CSDN 技术改变世界 创新驱动中国

Programmer 程序员

ISSN 1672-3252

2011年 2月刊

邮发代号：2-665 定价：15元

我的创业故事

精心筛选九位开发者的创业故事；
他们做过的产品，几乎可勾勒整个中国软件的产业链；
他们别出心裁的软件创富之路，会给您带来全新的视野；
他们与众不同的创业感悟，会给您留下触碰心灵的共鸣。

特别感谢 Contributors

（排名不分先后，以姓氏首字母为序）

Marty Cagan

感谢 eBay 前产品管理及设计高级副总裁 Marty Cagan 分享产品设计方面的经验，详见“TUP 专栏”。

岑文初

感谢淘宝开放平台主架构师岑文初为本刊撰文，详见“架构师拔龙”栏目。

陈榕

感谢上海科泰华提科技有限公司董事长陈榕接受本刊采访，详见本期“封面报道”栏目。

董洵

感谢 Trunk.ly 联合创始人董洵接受本刊采访，详见本期“封面报道”栏目。

高巍

感谢安卓爱普公司创始人高巍点评“社会化电子商务”发展的新契机，详见“评论”栏目。

高阳

感谢微软 ASP.NET MVP、技术作家高阳介绍 Dryad 系统的相关背景，详见“产品报道”栏目。

雷志兴

感谢来自百度 Web 前端研发部的雷志兴，介绍开源 JavaScript 框架“Tangram”的相关技术内容。详见“产品报道”栏目。

李建忠

感谢祝成科技创始人李建忠接受本刊采访，详见本期“封面报道”栏目。

林宇

感谢北京网秦天下科技有限公司 CEO 林宇接受本刊采访，详见本期“封面报道”栏目。

刘景龙

感谢百度基础架构部工程师刘景龙为本刊撰文，详见“技术”栏目。

马博

感谢 UCPM（中国产品经理联盟）发起人之一马博为本刊撰文，详见“TUP 专栏”。

马如悦

感谢百度基础架构部高级工程师马如悦对本刊“论文研读”栏目的支持。

Christopher Smith

感谢 RIM 公司黑莓开发平台高级总监 Christopher Smith 分享黑莓的最新技术与产品，详见“报道”栏目。

王昊

感谢 Esri 中国公司 CTO 王昊接受本刊采访，详见本期“对话 CTO”栏目。

王建硕

感谢百姓网 CEO 王建硕接受本刊采访，详见本期“封面报道”栏目。

王煜全

感谢 Frost&Sullivan 中国区总裁王煜全分享移动互联领域新观点，详见“评论”栏目。

徐乐乐

感谢 iDaily 创始人徐乐乐接受本刊采访，详见本期“封面报道”栏目。

杨立东

感谢北京暴风网际科技有限公司 CTO 杨立东接受本刊采访，详见本期“对话 CTO”栏目。

姚宏宇

感谢友友系统 CEO 姚宏宇接受本刊采访，详见本期“封面报道”栏目。

叶忻

感谢架势无线 CEO 叶忻接受本刊采访，详见本期“封面报道”栏目。

张辉

感谢网讯安卓副总经理张辉为本刊撰文，详见“程序人生”栏目。

张宴

感谢金山游戏运营技术中心技术经理张宴对本刊的支持，详见“架构师拔龙”栏目。

赵嘉敏

感谢译言、东西网创始人赵嘉敏接受本刊采访，详见本期“封面报道”栏目。

周爱民

感谢资深软件工程师、技术作家周爱民与读者分享架构师工具使用的心得，详见本期“我的工具箱”栏目。

人人都是创业者

本期我们介绍了几位技术人员的创业故事。

过去一年，我们在各种不同场合提醒大家，已经开始的技术大变革将深刻影响人类社会，同时也为技术人员带来了千载难逢的创业黄金机会。

相信会有很多读者认为，创业的成功概率那么低，风险那么大，是少数人干的事情，离自己很遥远。其实不然。

我曾经也认为管理是少数人的事情。然而，知道管理学大师德鲁克是怎么定义管理者的吗？他认为，在现代组织里，只要你需要自己作出决策，对某件事情负责，而这一事情又会实际影响到组织能否履行职责并取得成果，那么，你就是管理者。显然，球场上所有队员都是管理者。而在今天的现实生活中，由于专业化分工的深入，按这个定义，实际上绝大部分知识工作者都是管理者。

现在我要告诉大家，从现在开始，**人人都应该做创业者**。

为什么？

Rails之父David Heinemeier Hansson（DHH）和合伙人Jason Fried合著的《Rework》一书的开篇中说：

“这是全新的现实。今天，任何人都可以创业，过去远不可及的工具如今却触手可得。过去要价好几千美元的技术如今只要几块钱甚至能免费使用。现在一个人能干过去两三个人、有时甚至是整个部门的活儿。仅仅几年前还不可能的事情如今却变得易如反掌。

“你不必每周苦干60、80乃至100个小时，10～40个小时足够了。你无需耗费毕生的积蓄，或者承担巨大的风险。边工作边创业，就能维持足够的现金流。你甚至不需要办公室。而今你可以在家工作，或者与千里之外从未谋面的人合作。”

的确，开源技术和开放平台大大降低了技术创业的门槛，互联网使我们能够更容易地获取知识，结交志同道合者，并提供了强大而且成本低廉的营销渠道。那么，我们还有什么理由不去梦想，还有什么理由止步于梦想呢？

请注意，DHH和我都没有鼓励大家盲目地马上离开现在工作岗位的意思。事实上，创业并非都需要自力更生、白手起家，在公司里也需要创业：开发新产品、提供新服务、启动新项目、面向新市场，甚至用与以前不同的方法做事都是一种创业。虽然企业能提供更多的资源和现成的平台和保障，但是某种程度上，企业内创业所面临的困难更大，因为你会遇到惯性的巨大阻力，反对的声音和干扰因素也更多。

我们正面临永远在线的设备和服务所带来的史无前例的巨大变革，任何现在正在做、想当然的事情，都需要重新思考，寻找适应变革、面向未来的新的可能。也就是说，我们正进入事事皆创业的全新时代。

让我们整装待发，从现在开始，以创业者的心态为要做的每件事情赋予意义（而不是仅仅为了挣钱），提出口号，立即行动，找到赢利模式或者长期从事的理由，确定里程碑、前提，并分解任务（这是曾任Apple首席布道官的著名企业家盖伊·川崎在其经典的《创业的艺术》一书中总结的创业要点）。从现在开始，我们都是创业者。

邮件：liujiang@csdn.net
新浪微博：@刘江CE
欢迎大家交流反馈，欢迎投稿、批评、建议和挑错

Contents

Contents

勘误声明

由于我们工作失误，**2011年1月刊《程序员》**中，入选“**CSDN年度十大博客文章**”的《如何做一个出色的程序员》（**P79**）一文，正确作者应为岑文初，特此更正！

我们向作者岑文初先生致以真挚的歉意，由此带给读者的不便，我们也深为抱歉，并将以此为训，做好内容来源的审核工作。

现正确信息已在本刊官网发布（**http://www.programmer.com.cn/4609/**），敬请大家查阅。

《程序员》编辑部 **2011年1月20日**

2011年2月刊　总第220期

主管：中国社会科学院

主办：中国社会科学院文献信息中心

出版：《程序员》杂志社

网址：http://www.programmer.com.cn

国际刊号：ISSN 1672-3252

国内刊号：CN11-5038/G2

邮发代号：2-665

广告经营许可证号：京东工商广字0188号

读者服务部

Readers service Dept.

网上订购：http://dingyue.programmer.com.cn/

读者信箱：reader@csdn.net

地址：北京市朝阳区广顺北大街33号院1号楼福码大厦B座12层

Address: B-12th Floor Fairmont Tower NO.33 Guangshun North street, Chaoyang District,Beijing

邮政编码：100102

电话：010-64351436

传真：010-64348545

法律顾问：北京中润律师事务所 王杰

Law Consultant：Beijng Hengsheng Lawyer Firm

印刷：北京盛通印刷股份有限公司

Print: Beijing Shengtong Printing Co., Ltd.

出版日期：每月1日

Publication Date：the first day per month

零售价：RMB 15.00元　新台币 390元　HK $ 35.00（港、澳）　US $ 9.00（海外）

Retail Price：RMB 15, NT$390, HK $ 35.00,US $ 9.00

童话王国的文艺范儿科学家

——Peter Naur

■ 文 / 苏椰

在上一期名人堂中，我们介绍Alan Perlis时，谈到催生早期ALGOL语言的两场会议：1958年的苏黎世会议和1960年的巴黎会议。这两次会议，基本上是同一班人马，但在巴黎会议上，有一位新面孔为ALGOL60作出了巨大贡献，也为计算机程序设计语言的描述方式带来了一场革命。他的名字叫Peter Naur。

Naur，1928年出生于童话王国丹麦，29岁时，获得了天文学博士学位。其后他就职于哥本哈根天文台，在工作中，为了计算天文数字，他设计了丹麦第一台电子计算机DASK。Naur是一位富有文艺才华和情趣的科学家，我们检索一下他发表过的学术论文，就会看到，他的成果涉及了天文学、计算机科学、古典音乐、心理学等诸多领域，可谓是当之无愧的男子全能选手。

就在巴黎会议召开前不久，Naur进入了丹麦计算研究院工作，并为玻尔实验室授课，同时还当选为欧洲程序语言设计小组成员。尽管如此，Naur在这个时期的主要兴趣，仍然是天文学。对他来说，计算机只不过是一个用来研究天文的工具。

当初在苏黎世会议上，为了对ALGOL语言进行定义，Backus提出了一套以他命名的Backus范式，这是一个用来描述上下文无关文法的有力工具，学过编译原理的读者想必会对它十分熟悉。而巴黎会议的首要任务，就是对Backus范式进行改进，而这项任务，就是由Naur来完成的。为了使下一代ALGOL语言更加清晰，Naur对Backus范式进行了大幅度的简化，缩减了它的符号集，如今使用的Backus范式，实际上就是经过Naur改进之后的。

后来Donald Knuth指出过，应该将Backus范式更名为“Backus-Naur范式”，以强调Naur为其作出的巨大贡献。但是，谦逊的Naur却并不愿意这样做。在《人类行为——计算》一书中（这本书记载了Naur为计算机科学所做的大量工作），Naur说，我从来都不想把我的名字加进去，它一直都叫作“Backus范式”，我觉得这个名称就挺好的。

在巴黎会议结束时，由Naur对会议结果进行总结，他整理了讨论得出的ALGOL60全部特性，写就了一份后来名扬江湖的《算法语言ALGOL60报告》。

这个总结工作，充分体现了Naur的个人才华以及Backus范式的力量。在这份报告之前，编程语言不是由语法定义来描述的，而是由编译器代码来描述的。你要学习一门语言，就要去看它的编译器的源代码，既冗长又混乱，非常不便于学习，也不便于人们之间进行交流。这就好比你买了一部手机，发现附带的说明书很有特色，是这个手机的全部电路图，请你根据这些图来学习如何使用手机。当时学习编程，就是这么尴尬。

但Naur改变了这种方式，他使用Backus范式，对ALGOL60的语法进行定义，然后用自然语言精心措辞，短短的17页报告，简洁优雅至极。这份报告后来成为计算机科学史上的一个

名作，它完整地描述了ALGOL60语言的全部特性，却不依赖于任何机器细节，因此非常便于人类互相交流——编程语言第一次拥有了“使用说明”而不是“电路全图”。Naur的这项工作，改变了此后描述编程语言的方式，一直影响至今。

而他本人，也因为这场巴黎会议一鸣惊人。这位来自童话王国、热爱古典音乐的天文学家，站在这里告诉世界，计算机科学也可以很美的。

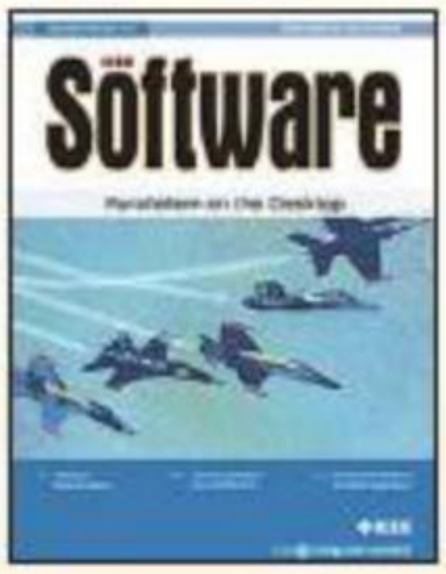
在ALGOL60报告发布的45年后，2005年，77岁高龄的Peter Naur得到了迟来的ACM图灵奖。ACM给Naur的颁奖词是：

“授予Peter Naur图灵奖，以表彰其在定义ALGOL60程序设计语言方面的先驱性工作。”

虽然这份奖励姗姗来迟，但是以Naur的淡泊和平和，我们不难想见，他不会太介意的。

IEEE Software
2011.01

桌面上的并行



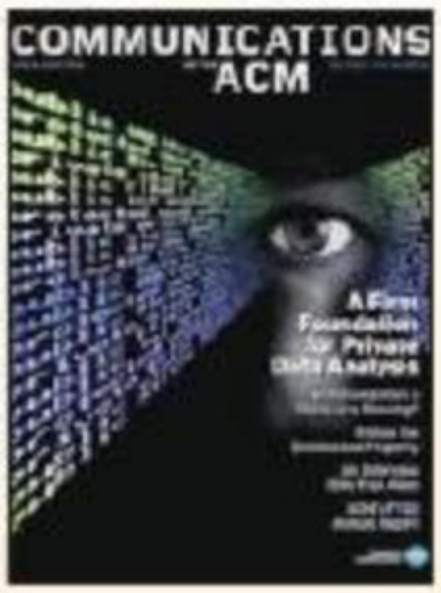
计算机行业正在经历一个重大转变：因为发热问题，通过更高的时钟频率提高单个处理器的性能已面临技术的极限。幸运的是，摩尔定律仍然有效，所以芯片制造商使用晶体管通过并行提高多/众核处理器的性能。然而，这需要并行编程来充分发挥这些处理器的潜力。因此，大量的开发者致力于将桌面应用程序并行化，包括浏览器、商业应用、多媒体处理以及其他特殊领域的具体应用。这很可能会导致对桌面软件有史以来最大规模的重构。要做到这一点，必须将系统工程原理应用于对性能优先的应用和环境的并行化。鉴于这一开发趋势的进展，本期的《IEEE Software》特别关注了在并行化桌面应用时，有关编程方法、工具和库的特殊问题。

在《并行的一种重构方法》中，作者提出了一个重要问题，即如何利用现有的串行代码生产出并行代码。他特别介绍了为线程安全、吞吐量和可扩展性而进行的不同的Java重构，并且提出了一个半自动的工具集，帮助软件工程师更快地完成工作。《用英特尔线程构建模块进行多核桌面编程》一文中，作者给出了一个能够被广泛应用的并行库的例子，并解释如何使用它。像这样的并行库特别有助于减少现实中并行桌面应用的开发时间。此外，作者还解释了该库的内部结构和设计理念。

《联合的力量：从多线程编程到GPU编程》一文展示了桌面并行化的最新发展——多核处理器和图形处理器的同时使用。作为入门，作者概述了这种新一代的异构并行应用的技术细节，另外还讨论了在算法设计、内存传输及数据大小、控制精度处理和计算精度上的最佳做法。

Communications of the ACM
2011.01

技术前沿



本期的《Communications of the ACM》聚焦于社交媒体、隐私、虚拟化、并行计算等问题。

《Opensocial：让互联网上的社交应用成为现实》一文介绍了OpenSocial API套件。在该套件基础上，开发者创建的应用程序能同时使用来自多个不同社交网络的内容。

差别隐私（Differential Privacy）的蓬勃发展，在于它是自然的，不囿于特定领域，并且与其他领域有卓有成效的相互作用。在传统的密码安全观点看来不适当或不切实际的情况中，比如隐私数据分析，这种灵活性有希望给出一种原则性的解决方法。《隐私数据分析的坚实基础》一文中就首次将隐私数据分析建立在了数学基础上。

从资源利用不足到数据中心优化再到减少碳排放，虚拟化常常被吹捧为解决这些挑战性问题的方案。然而，复杂性和系统管理困难度的挑战，形成了虚拟化的隐性成本，常常被忽视。要从虚拟化中获得收益，要求企业解决可扩展性限制，改革传统的运作方式和管理性能，实现史无前例的跨团队合作。《虚拟化：祝福还是诅咒？》对管理大规模虚拟化充满的隐含挑战进行了探讨。

云计算意味着将数据托管给信息系统，在“云中”的远程服务器上，由外部人员管理。云计算引发了隐私和保密问题，因为服务提供商一定能够访问所有数据，并可能无意或故意地泄露或者未经授权使用数据。《近在咫尺的云计算隐私担忧》以基于云的会议管理系统为例，对隐私问题进行了探究，强调了可能产生的不良后果，并为问题解决指出了方向。

Technology Review
2011.01

未来电视与社交网络

本期的《Technology Review》封面文章是《探寻电视的未来》。2010年5月20日，谷歌电视首次公开展示。这款软件旨在提供一种简单的方法，让人们在更大的屏幕上，访问普通电视频道和互联网上浩如烟海的内容——在互联网时代重塑电视。《探寻电视的未来》一文考察了谷歌电视的诞生和发展过程，探讨了电视行业的未来发展方向。指出虽然谷歌和硅谷的极客已经开始致力于革新有70年历史的电视产业，但要重塑电视行业并非易事。

《观察观众》一文指出，在过去的几十年中，科技产业存在着一种错觉，即消费者会喜欢他们的电视机变得像台电脑，因此做出很多吃力不讨好的努力。虽然包含了很多令人难以置信的智能技术，但效果并不显著，很多这种设备最后都进入了坟墓。作者任职于英特尔公司，他们自身就有过多次失败的尝试。她指出，要让电视变得更智能，首先要了解是哪一点让电视成为了我们最热爱的一件物品。

Facebook已经将触角伸到了全球各个角落。拥有5亿用户，美国之外的用户就占总数的3/4。但《结交朋友》一文指出，Facebook仍有很大的发展空间。它才刚刚开始涉足印度和巴西这些人口大国，遭到世界上最大的互联网市场的屏蔽，在日本和韩国仍然处于劣势。该文认为Facebook这6年来能够轻松地让用户自然扩展，要归因于网络效应的强大力量。但是现在，在像日本、韩国这样更难被西方公司攻克的市场中，它必须加倍努力，才能立足。

相对于Facebook的一路高歌猛进，第二人生就没那么幸运了。曾几何时，第二人生营造的网络虚拟世界看起来像是下一个大热门，只不过现在很大程度上已经成了明日黄花。而无论是拥趸还是诋毁者，都无法理解这一点。《严肃的游戏》一文，对第二人生的兴衰进行了探讨。

观点

@tualatrix

为什么那么多开源软件开发者使用Mac呢？这个问题有很多答案。不过可以看看2005年时，Linus Torvalds回答他为什么在用Mac：http://goo.gl/nLEPc。

@nnfish

Larry Wall在平安夜的博客上回顾了Perl语言的历史，及其与黑客文化之间千丝万缕的联系。他称，Perl不仅仅是一种技术，它还是一种文化，黑客文化。

@middlefeng

这个是大牛Tom Forsyth的观点，http://bit.ly/fG17GQ。 @marchliu我的另一个观点是在广泛的集成环境下，如果对最终品质要求比较高的话，两层栈（内核+App）比三层（内核+VM+App）更容易解决问题。

@mitchjoel

数字内容支付真的有希望吗？@markwschaefer认为有，但是……http://bit.ly/e3JCSr。

资源

@Typeisbeautiful

2010年最糟与最好的标识重设计排行榜，GAP与普华永道分别荣登榜首，http://is.gd/k4JhL ，http://is.gd/k4JFY。

@CatChen

Tangram前端库通过Github开源了，http://dlvr.it/C42fC。

@DigitalSonic

不知道Voldemort的，可以看看这个Slide：Hadoop and Voldemort，@LinkedIn http://www.slideshare.net/hadoopusergroup/hadoop-and-voldemort-linkedin。

@ucifierya

2010年的10个最佳数据可视化工程，http://j.mp/h2ndDs。

@diamondtin

http://appstorm.net挺好的，看看人家的深度评测，找找软件灵感。我发现了一堆自己想用，自己想做还没做，人家已经做得超级好的App。

@zhaozexin

第5届D2的志愿者们非常敬业，嘉宾PPT、视频、反馈表统计都按时出来了！ 感谢他们， http://www.d2forum.org/。

文章

@gaochunhui

JavaOne 2010感想，草稿篇：本来是想专注记录JavaOne北京2010的见闻，无奈北京站比起旧金山的原版缩水太多，不拉上原版感觉很不够力，所以把JavaOne 2010整体拿来写个感想系列，http://bit.ly/gxD8gk。

@hongqn

Beansdb卷土重来，http://www.douban.com/note/122507891/。

@audreyt

“小白与Perl 6：华丽的冒险”，http://pugs.blogs.com/pugs/2010/12/yule.html。

@marchliu

如果你还不明白什么是递归，就读这个句子，http://fleurer-lee.com/lyah/recursion.htm。

@plinux

新博文：PostgreSQL和MySQL的对比，第1部分：表组织 http://bit.ly/fm8rEe。

@lidaobing

HTML5设计原理：为之漫笔， http://goo.gl/eBX5R。

@bang590

博客：移动浏览器的viewport ， http://blog.webang.net/tech/1572/。

（网址链接请访问杂志官网，www.programmer.com.cn）

CSDN最受关注文章 TOP10

（2010年12月26日~2011年1月26日）

1.	将会成为2011年主流的七大IT技术
2.	马化腾兑现诺言：开放QQ空间第一步
3.	外媒评2011 年十大必“败”平板电脑
4.	美新闻周刊评十大创新企业：微软居首，中国崛起
5.	2010年走到转折点的IT人：他们被迫改变
6.	一个应聘者的Groupon面试经历：疯狂的招募
7.	1月编程排行榜：Python成为2010年度语言
8.	评论：谷歌移动呼之欲出
9.	乔布斯病休，谁将成为接班人
10.	微软即将宣布下一代Windows系统：支持ARM架构

JavaEye最受关注文章 TOP10

（2010年12月26日~2011年1月26日）

1.	20款优秀的可以替代桌面软件的Web应用
2.	用好IE9浏览器必须要知道的九件事
3.	开源项目eRedG4_V1.03.1发布了
4.	八个工具助你快速搭建移动版网站
5.	十个jQuery图片画廊插件推荐
6.	Facebook超越Google成为访问量最大的网站
7.	10个能帮你提高工作效率的CSS工具
8.	入驻百度应用开放平台开发者已超过1500名
9.	流行的开源网站建设工具Drupal 7发布
10.	解密Opera是怎么赚钱的

状态机工具

主持人：
潘加宇，UMLChina首席专家，潜心研究和实践UML/UP相关技术的应用。

软件工程与管理

Quantum Leaps发布了一款免费的状态机建模工具QM（QP Modeler），用于设计嵌入式实时应用并无缝地在QP（Quantum Platform，量子平台）实现。

QP是一套状态机的实现框架，支持的RTOS和处理器非常广泛，已经在消费电子、医疗、工业、无线、网络、国防、机器人、汽车等许多领域的实时嵌入式开发中使用。QP的作者Miro Samek的著作《Practical UML Statecharts in C/C++》更是学习状态机建模和实现状态机不可不看的深度教材，第一版在2004年已经由国内出版社引进，第二版中译本也即将在国内出版。

Miro Samek长期从事嵌入式软件开发，曾就职于GE Healthcare、Novariant、Global Locate。多年来，他一直专注于状态机建模和实现的研究，花了很多精力来打磨QP框架、打通状态机和下层RTOS的通路，这是非常难得的。

这次Miro Samek决定开发QM这样的图形建模工具，可以看做是QP向上走、打通和UML建模工具的通道。Miro Samek想走一条和之前的状态机工具有所不同的道路，用他的话讲就是“不需要和工具作战”。QM是先有QP这个实现框架，再有图形建模工具，图形工具建立在代码框架的基础上。

其他状态图工具如IAR VisualState（最新版本6.3.2），还有IBM Rational Rhapsody（最新版本7.5.3）走的是自上而下的路线，也支持状态图的验证和测试，生成的代码和RTOS整合。

现在使用的状态图是由David Harel在1980年代发明提出的，增加了层次、并行等元素，改进了之前常规FSM（有限状态机）复杂性所带来的缺陷。状态图是面向对象建模中非常重要的要素。对象身上发生的行为导致对象属性值的变化，属性值的变化又导致行为的变化，这种不同行为之间精妙的互动关系，通过状态图可展现出来。

可以这样说，如果没有把核心的逻辑封装到关键类的状态机中，只能说这样的面向对象建模是“假面向对象”。通过状态图可以帮助开发人员确定一个类合适的责任到底是哪些，往往可以起到缩窄类和外界的接口、加强封装的作用。

遗憾的是，状态图在实践中经常被误用，误用频率恐怕和用例有一比。常见的误用例子是：不理解状态图的本意，为不存在状态的类画状态图，导致把状态图当成活动图使用，其实状态图和活动图相比，顶点和边所代表的意思是相反的。

移动开发和智能设备将是未来几年的开发热点。客户对移动设备和智能设备上的应用质量要求更高，有必要通过状态机来对每个类作精细的建模，生成能直接使用的代码，尽量减少人工编码引入的偶发错误。而且，这也是有可能的，毕竟移动设备和智能设备上的应用中，类的个数并不多。

本月热点

事件

首届全国大学生UML建模创造力大赛结束

从获奖作品的名字：幼儿园CAI教学系统、租房社区系统、火车票实名制解决方案等，可以大致看出现在的社会关注点。

资源

QM下载

QM主要用于设计嵌入式实时应用并无缝地在QP得以实现。下载地址如下：

http://www.state-machine.com/downloads/index.php#QM。

David Harel的状态图文献

1984年David Harel发表的关于状态图的文献：*Statecharts: A visual formalism for complex systems*。

http://www.wisdom.weizmann.ac.il/~harel/papers/Statecharts.pdf。

David Harel的演讲视频

2010年David Harel在Federated Logic Conference （FLoC）上缅怀图灵奖获得者Amir Pnueli的演讲视频。

http://www.youtube.com/watch?v=KJ9WaUa_4m0。

状态机建模工具sinelaboreRT的2.24版本

Peter Mueller也发布了状态机建模工具sinelaboreRT的2.24版本。新版本除了继续支持原有的商业工具如UModel、Magic Draw、EA等，还增加了对开源UML工具ArgoUML的支持，这样，开发人员可以利用开源工具来建模，通过sinelaboreRT生成代码。这款工具的特点是支持从状态图生成Objective-C和nesC代码。

主持人：
王翔，软件架构师，主要研究方向为XML、.NET、领域设计和PKI应用。工作之余喜爱旅游、写作和烹饪。

从展会看DBA的职业成长与发展

数据库

本月热点

● 新品发布

甲骨文发布新的Sparc T3芯片搭建的服务器系列——SuperCluster

甲骨文官网上介绍了其最新的服务器系列。我们要知道，不一定大块头才有大智慧，应该关注SuperCluster以及它对大型企业应用的影响。

http://www.oracle.com/us/corporate/press/192208?rssid=rss_ocom_pr。

MySQL GA 5.5发布

MySQL官网上介绍了曾经的宠儿——MySQL的最新发展，它的未来我们很关注。

http://dev.mysql.com/。

● 技术会议

微软TechED 2010中国大会

微软官网上介绍了有关2010年微软年度技术大会，回顾、总结、展望。

https://partner.microsoft.com/China/40146927。

Oracle OpenWorld中国大会

甲骨文官网上介绍了有关2010年甲骨文年度技术会议中人员、资讯云集的盛况。

http://www.oracle.com/cn/openworld/en/oow-agenda-en-071652.html。

● 推荐资源

三家NoSQL厂商推出NoSQL介绍教程

CouchDB、MongoDB和SimpleDB三家数据库厂商分别推出了NoSQL介绍教程。但是NoSQL想说爱你不容易，不是我不熟悉，而是你们往往不走寻常路。

http://www.readwriteweb.com/hack/2010/12/3-new-nosql-tutorials.php。

2010年年末，国内顶级技术盛会云集，其中微软的TechED和甲骨文的OpenWorld在技术社区中影响更甚。不谈具体的新技术、理念和设备，仅从DBA自身职业成长看，值得思考的地方很多。

在TechED中，围绕SQL Server的数据管理、商务智能的课程比例不小，而甲骨文OpenWorld围绕Essbase和Oracle数据库的课程比例更多。尽管课程侧重不同，但从DBA的角度看这些介绍均预示着人工平均维护成本、管理边际成本都是递减的，DBA通过浏览器可以“望闻问切”的系统规模愈甚。而且，企业信息化的发展可以打破自己技术投入、技术队伍的限制，通过数据云平台实现跨越式发展。打个未必恰当的比喻，以前信息化是自己买车、养车，然后到达目的地；后来有了IT服务外包和托管后相当于直接打车到目的地；而现在有了数据云平台就和坐地铁、乘公共汽车一样。

生产率的提高和简化对于身处职场的DBA而言不能不说是威胁，DBA似乎越来越“被离开”IT团队的中心了，实际是否如此？否。如果说摩尔定律昭示了这个行业令人可喜的向上生长一面，那么这个增长速度恐怕还要“输给”信息环境复杂化的步伐，当企业把业务思考寄托在信息环境之后，它就像决策者的思维一样，每个系统作为独立的神经元，相互间纵横交错越来越复杂，这绝不是一根服务总线能够掩盖的。如果说以前的DBA处置的是几个数据库，那么现在他面临的是一群数据库，以及连接这些数据库的信息系统、队列、服务接口、安全机制和基础设施平台。于是DBA的“A”可能要从Administrator变成Architect，而对信息化全局性规划能力的要求是DBA必须面临的挑战，一方面是快速发展的各类技术，个个都是“双刃剑”；另一方面是同样快速变革的市场和业务环境，DBA要做的就是把它们连在一起。不过这些所谓的“高层的”要求并不代表全部，正如大家熟知的，一个庞大的信息系统上线后往往由于一两个功能点导致全局问题，这又要求DBA有深入的洞察力。

说到“新”技术，云平台和虚拟化应该算这些盛会上的明星，DBA要从三个层次看待它们：第一个层次，将它视为一个技术领域，用于自己企业的信息化建设，提升工作能力；第二个层次，将它们视为合作伙伴，明确地将企业的信息化内容进行切割，明确哪些是“放在外面的”、哪些是“放在里面的”，提升自己的工作范围；第三个层次，可能的情况下将设计、建设和运维这样的云平台作为自己职业发展的下一个目标，提升自己的市场高度。

但这样的要求对于DBA是否过于苛刻呢？有些，但不完全。毕竟，DBA是DB的“A”，专业领域还是数据库和现在越发流行的数据仓库部分，其他领域他只要“善假于”即可，所以参加展会应“风物长宜放眼量”于行业、于自身。

开放 API，让企业管理软件连起来

主持人：
邢波涛，现任北京新软孚信息技术有限公司技术负责人。关注SaaS管理软件和B2B、B2C电子商务的融合。新一代电子商务和SaaS管理软件积极的实践者。

企业应用与行业软件

上个月，跟淘宝的人一起去拜访淘宝三钻以上客户，跟他们聊起来，淘宝开放平台在2011年下半年将要支持和用友、金蝶的接口了，淘宝的进销存数据可以直接导成用友、金蝶的财务数据，淘宝也即将开通短信开发平台，供基于淘宝开放平台的开发者调用。

事实上，开心网、人人网、新浪微博，甚至腾讯网，也都陆陆续续地推出了自己的开放平台和开放API，以供开发者基于它们的平台做二次开发。这就为我们的管理软件数据，在这些互联网公司之间的网站流转起来，创造了可充分发挥的空间。

所谓“单一做到卓越，整合创造价值”。苹果公司仅仅凭借iPod、iPhone、iPad寥寥几款产品，短短几年内，东山再起，就做到世界市值第一，引导消费者消费潮流好几年，单一产品做到了卓越品质。而我们的管理软件，对中小公司而言，个人判断不能再像SAP、Oracle、IBM那样凭借业务专家的模式玩儿了。先说别人有病，再派自己的咨询人员给人诊断，然后再由售前画个蓝图（Blue Print），大不了再做个证明自己能力的POC原型，就千万甚至上亿美元开始收钱了。小公司没有那么好的大夫，也没有那么好的业务专家，更不可能具有SAP、IBM那样的权威性。对于中小管理软件公司来说，做项目，一则长期没有保障；二则针对每一个项目，就像搞家庭装修一样，最后的结局都是客户基本不满意，回款太难要，很难长大。这也就是为什么外包企业最近十年一直红红火火，反观做管理软件的却基本偃旗息鼓的原因。

而互联网和电子商务公司最近两年，风水再次转好，为互联网公司服务，为电子商务公司服务。在以前管理软件生成的业务数据在用户自己的平台、电子商务公司网站、互联网网站之间流转起来，让业务数据成为互联网营销数据。个人认为，套用一个网络流行话语：“换个思路天地宽啊，天地宽。”这样，就使得我们的管理软件和互联网无缝地结合。由于这些大网站之间API的开放，使原先一个个数据孤岛统统连接起来成为可能。也许很多人会很担心业务数据的安全性，是否会被窃取或被竞争对手获得。其实，哪些数据可以暴露出来，是需要用户自己规划出来的。对于商品的基本信息，例如商品公开报价、简介说明，我们完全可以通过新浪开放平台，同步到新浪微博上去。目前，新浪微博上，已经有一些整合很多商家的商品打折信息的插件了。而我们的管理软件，这些数据已经被客户录入到自己的内部信息系统中。如果把这些数据开放出来，通过新浪微博、开心、人人网的开放API，把它们互联互通起来，那么原先我们的基于用户内部网络的管理软件，就会和外部互联网互联互通起来，我们的管理软件通过这种思路，会不会重新找到另外一种商机呢？

本月热点

● 新品发布

亚马逊Android应用商店面向开发人员开放

2011年1月6日，亚马逊面向开发人员开放了Android应用商店，邀请开发人员为该商店开发应用。亚马逊Android应用商店将于今年晚些时候面向普通用户开放。又是一个管理软件结合电子商务、移动互联网连接起来的商机。

Cassandra 0.7即将发布

Cassandra于元旦发布了Cassandra 0.7.0的RC4版本，RC4之后，距离正式发布日期就不远了。Cassandra已经是最重要的NoSQL数据库之一了。

● 事件

Google 贡献两个项目给 Eclipse 基金会

这两个项目分别是WindowBuilder，一个基于Eclipse的Java GUI设计器，以及CodePro Profiler，开发者可以使用该工具来检查应用的内存泄露问题和性能问题。据称这两个项目大约值500万美元。Google将开源 WindowsBuilder，这是基于Swing和SWT的设计器，也是Eclipse的Google插件的一部分。

主持人：
姚磊，Microsoft Dynamics CRM MCP，多家企业信息化商业解决方案项目经验。熟悉IT规划与需求工程与项目管理。

ICC：电子商务公司核心

企业应用与行业软件

本月热点

新品发布

甲骨文推Oracle Cloud Office 1.0和Oracle Open Office 3.3

2010年12月16日甲骨文公司宣布，推出两款全面和基于开放标准的办公用生产率套件Oracle Cloud Office 1.0和Oracle Open Office 3.3，这两款产品面向桌面、Web和移动设备，可帮助用户显著提高效率、降低成本并在整个企业范围内实现更有意义的创新。

事件

用友云战略正式浮出水面

管理软件提供商用友日前正式发布云战略。用友实施云战略将结合自身的优势，通过云计算技术、服务和商业模式的创新，为企业和机构提供丰富的企业云服务、行业云服务，帮助客户迈向云端企业，让企业漫步云端。

会议

2010年度中国信息化领袖峰会在沪举办

2011年1月6日，以“转型升级下的中国企业信息化价值”为主题的2010年度中国信息化领袖峰会暨商用软件评选颁奖大会在上海召开。超过300位嘉宾汇聚一堂，来自主办方、政府组织、行业协会、全国各地工商企业的代表，来自公司决策、信息化、财务、采购、生产运营等职能部门的负责人，SAP、金蝶、Infor、东软、QAD、IFS、华胜天成、远光、源天软件等国内外近50家信息化解决方案供应商出席了本次会议。

最初的呼叫中心只是个简单的电话系统，客户通过电话来获取资讯信息，企业通过它来联络客户、推广产品。随着计算机技术的发展，呼叫中心发生了根本变化，功能日趋丰富，向着语音合成、IP、多媒体、统一消息等方向发展。

而现在随着中国网民数量的激增，在线电子商务交易也是得到了快速的发展，如淘宝、当当网、京东方、新蛋、拍拍等。广大网民在线消费时主要是通过网络与企业进行沟通的。所以ICC（Internet Call Center，网络呼叫中心）就应运而生了。

ICC与常说的因特网呼叫中心在实质上是一样的。它是将呼叫中心与Internet集成为一体，用户可以通过不同的终端和接入方式访问呼叫中心，获取所需的信息和服务。

它不是简单地把互联网信息提供给呼叫中心，而是将呼叫中心与互联网集成为一体。用户可以从Web站点直接进入呼叫中心，用点击按钮的方式实现与对方通话。它再也不是一个传统概念上的CTI技术了，在这里，它可以有线呼入、移动（无线）呼入、IP呼入。

由于ASR、TTS等技术的采用，可以实现把语音信息自动变成文字、文字自动变成语音，完全看用户如何方便。

当然远端可以用IP电话，也可做文本交互（如白板功能），自然一切Internet上的功能都可结合为一体共同使用，如E-mail、IP传真。由于IP电话、IP传真、E-mail的价格便宜，使得这种呼叫中心为大的跨国公司建环球服务中心成为可能，用户不用800和400号也可以全天候呼叫，企业减免了电话费用负担。

DW（DataWarehouse，数据仓库）是一种近年来发展起来的用于决策系统的技术，在国外很多规模较大的呼叫中心都引入了这一技术。它的主要功能是对用户的资料、市场的资料、业务、财务、成本、利润各项有关数据进行统计分析，供公司领导者做公司决策之用。有远见的企业家在建立呼叫中心时一定要把DW看成是ICC必不可少的一部分，而且是非常重要的一部分。

电子商务公司的所有信息可以自由往来于PSTN网、IP网，所有的信息会自动记录下来，并不断积累改进，形成为用户提供个性化服务的基础，这些系统不仅可以提供信息服务，也可以完成电子交易。

大量的运行数据通过DW作出记录、统计、分析。领导者不断根据所获得的结果做决策，改善服务、改善经营。

这样一个新的完整的基于CTI技术的网络呼叫中心已为电子商务构架了完美的模式，在未来的一年乃至更长的时间里，它将会成为电子商务的主题、平台、核心以及灵魂。因此ICC技术已不是传统概念上的CTI，也是未来从事电子商务平台发展业务的一项重要支持技术。

跨平台动态语言 Python 新动态

主持人：
俞黎敏，IBM高级工程师，满江红开放技术研究组织核心成员。开源爱好者，在各大技术社区为推动开源和敏捷开发积极摇旗呐喊、加油！

动态语言

在TIOBE 2011年1月开发语言排名当中，Python荣获2010年度大奖，这是Python继荣获2007年度大奖之后，再次赢得此殊荣，按时下最流行的说法真是很给力啊！

“There is only one way to do it, the right way”，这是Python的创始人Guido van Rossum的豪言壮语。Python从语法层面杜绝个人风格对代码可读性和可维护性的伤害。Python是一种面向对象、直译式计算机程序设计语言，也是一种功能强大而完善的通用型语言，已经具有十多年的发展历史，成熟且稳定。这种语言具有非常简捷而清晰的语法特点，适合完成各种高级任务，几乎可以在所有的操作系统中运行。目前仍是两个主力的版本并存，2.x系列目前的产品版本为Python 2.7.1，3.x系统的则为Python 3.1.3，3.x何时完全更替2.x，还需要拭目以待。

自2010年1月以来，Python获得了较大的市场份额增长，为1.81%，是增长速度最快的。这比时下盛行的“苹果迷们”手中持有的设备所用的Objective-C（1.63%）处于领先的地位。Objective-C在很长一段时间里由于苹果的iPod、iPhone、iPad产品的普及而受到极大的追捧，然而2010年的最后几个月它的市场份额有很大的下滑。Python已经成为脚本系统标准，以前是Perl，但现在更为不同类型的应用领域使用。比如Python用在非常流行的Web开发中，特别是与Django框架的组合。因为Python简单易学，越来越多的大学教授也开始使用Python编程语言进行教学。

Python语言本身有很多种优点，也表现得很不错，但是在我们身边采用Python的公司还是比较少，认识它的价值的公司更少。Python除了在后台脚本动态运行、快速测试开发给我们带来许多优势之外，最近几年随着Python Web开发框架的快速发展，Python在Web开发领域的运用也越来越多，国外也出现了一大批采用Python开发的新网站。这当中有著名的文件分享云平台dropbox.com、内容阅读互动评论网站reddit.com、好友动态信息推送friendfeed.com，其中Reddit代码还是开源的，Friendfeed则是采用自己写的Tornado。

Python2.x和Python3.x差异性比较大、前后版本不兼容，虽然有相应的2.x到3.x工具进行转换，但是并不能解决所有的问题。目前Python3.x尚未完全普及，而且很多第三方的类库也没有开始支持Python 3.x。考虑到前后版本的不兼容性，会让开发者对采用Python开发项目产生相当大的顾忌。而在Web开发领域，WSGI（Python Web Server Gateway Interface）v1.0也是不支持Python 3.x，支持Python 3.x的Python Web3 Interface（PEP 444）还处在草案状态中。当然，随着Python 3.x的普及与Python Web框架的不断发展与成熟，Python有着很大的上升空间，未来是否能够超过PHP则有待时间来证明！

本月热点

● 新品发布

GAE SDK 1.4.1正式发布

GAE（Google App Engine）可以让你在Google的基础架构上运行网络应用程序。部署在GAE上面的应用程序易于构建和维护，并且可根据访问量和数据存储需要的增长轻松地进行扩展。使用GAE，将不再需要维护服务器：只需上传已开发好的应用程序，它便可以为用户提供服务。

GAE支持以几种编程语言编写的应用程序。通过GAE的Java运行时环境，可以使用标准Java技术（包括JVM、Java Servlet和Java编程语言，或使用基于JVM的解释器的任何其他语言，例如JavaScript或Ruby）构建应用程序。GAE还提供一个专用的Python运行时环境，该环境包括一个快速Python解释器和Python标准库。Java和Python运行时环境构建为确保应用程序快速、安全运行，并不受系统上的其他应用程序的干扰。

GAE轻量级Groovy框架：Gaelyk 0.6发布

Gaelyk是一个专门针对Google App Engine Java而进行设计的轻量级Groovy工具。如果有打算开发基于Google App Engine Java的网络应用程序，可以考虑使用Groovy和Gaelyk来进行快速的开发，Gaelyk可以让你采用Groovy编写应用程序并部署到Google App Engine上面。Gaelyk提供Groovlet和Groovy Template Servlet方便进行快速开发，简化了Google App Engine SDK的使用，并提供了更加简洁的和强大的快捷方式，比如使用数据存储、图像服务、URL服务、邮件收发、Jabber消息等。Gaelyk 0.6发布，该版本的重大改进：支持GAE SDK 1.4.0和Groovy 1.7.6、在初始化插件系统时使用了Servlet Context Listener、开始支持异步数据存储等。

主持人：
高昂，中国标准化研究院助理研究员，从事信息技术标准化研究工作。关注开源社区，也是OSGeo中国和InfoQ中文站成员。

跨越JVM与.NET CLR的Fantom语言

编程

本月热点

新品发布

Foursquare两款开发工具开源

1月17日，LBS（基于位置服务）的代表Foursquare将两款内部开发工具开源。它们是用于MongoDB查询的DSL（特定领域语言）Rogue和用于iOS开发的Full-Loaded。其中Rogue是一个类型安全的内部Scala DSL，比MongoDB自身提供的查询语言更具表达力。此外，Foursquare还开源了三个其他种工具：foursquare-palmpre、asi-http-request、foursquair。

事件

Django 1.3发布计划公布

开源Web应用框架Django在第一个1.3 Beta版本发布后，又在官方站点公布了Django 1.3正式版本的发布计划。Django 1.3加入新的框架，便于让开发者使用基于Class的视图。Django 1.3还支持Python的logging模块，并提供了Python单元测试工具unittest2的支持。

资源

Fantom vs Scala

Fantom语言的作者针对Fantom与Scala语言的特性，进行了全方位的比较，以帮助开发者选择合适自己使用的编程语言。

http://fantom.org/sidewalk/topic/675。

FantomIDE

FantomIDE是基于NetBeans Platform开发的Fantom编程环境，可以单独使用，或者作为NetBeans IDE的插件使用。

http://fantomide.colar.net/。

在JVM平台并入Oracle产品线后，Oracle将对JVM平台进行多项改进和调整。内容包括增强平台的模块化和集成性等特征、为JavaSE增加多核处理支持、在JavaME中增加多点触摸支持等。

与此同时，Oracle开发部副总裁Adam Messinger在Qcon开发大会上透露，Oracle计划在提供免费版JVM的基础上新增收费版本，在收费版JVM里整合HotSpot JVM与JRockit JVM，为高端用户提供功能更为强大的解决方案。

对开发者来说，无论Oracle策略如何，JVM平台依然有着巨大的吸引力。依托于JVM平台的动态语言，始终保持着蓬勃的生命力。在这些动态语言中，能够同时兼容JVM和.NET CLR通用语言运行时的Fantom，是其中极具特色的动态语言。

Fantom最初被称为Fan语言，由于命名为Fan不易检索，开发者在2009年11月将当时新发布的版本定名为Fantom语言。Fantom使用与BSD 类似的Academic Free License 3.0 开源协议授权。

Fantom的开发者将其设计为一门实用且有趣的脚本语言，来解决Java和C#编程中实际存在的问题，并降低开发者编码的难度。Fantom语言最大的特点是它的可移植性，Fantom代码能够无缝地在JVM和.NET CLR平台之间进行迁移。为了保证平台间的顺利移植，Fantom代码首先被编译为fcode，即一种字节码表示法，之后转换成为Java字节码或IL解释语言，转换的过程在程序运行时完成，开发者可以以单个文件的形式在虚拟机中部署Fantom代码模块。

对于JVM和.NET CLR平台切换时的API调用，Fantom给出了简便易用的解决办法。Fantom提供了一组与Java和.NET API配合使用的API接口供开发者使用，来屏蔽不同平台间对于系统API调用的差异。

当然，Fantom语言的可移植特性不仅限于Java和.NET平台，它还支持将代码编译为JavaScript脚本供开发者在浏览器端使用。Fantom语言从一开始就为跨运行时的编程而设计，在未来的规划中，Fantom语言还将支持iPhone的Objective-C语言，以及LLVM集成编译环境或集合了Python与Perl语言优势的Parrot语言。

在Fantom编程支持方面，开发者可以使用基于NetBeans Platform的IDE环境FantomIDE进行编码和调试，IDE包含开箱即用的Fantom环境、语法和词法解析使用开源语法分析器ANTLR设计；FantomIDE还提供了JavaScript、CSS、HTML、XML等脚本支持以及Subversion和Mercurial等管理工具。

JVM平台从不缺乏优秀的动态语言，Groovy、JRuby、Jython、Scala都是其中的佼佼者，不同语言的风格类型、运行速度和工具支持也各有千秋。Fantom以其跨运行时的特性和先进的设计理念，将来是否能脱颖而出得到开发者的认可？让我们在实践中得出答案。

可爱的 Mono

主持人：
涂曙光，微软平台技术专家，专注于.NET、Web、SharePoint、Office等技术领域。现任职于中国惠普有限公司。

.NET

如果你是.NET开发人员，至少听说过Mono这个名字。简单来说，Mono是.NET的一个开源实现，它包含了一个类似CLR的运行平台和C#语言的编译器。最重要的是，Mono是开源且跨平台的，它弥补了.NET程序只能运行在Windows平台上的缺点，为.NET开发人员打开了通向其他平台的一扇窗户。

基于Mono，已经发展出了不少周边的工具和扩展。例如MonoDevelop就是一个开源的C#开发工具，它能运行在Linux和Mac OS X上，Moonlight则是一个Silverlight的开源实现。从发展的情况来看，Mono并未如很多人预料的那样被人淡忘，相反，Mono社区越来越活跃，也有越来越多的.NET开发人员开始关注Mono。

智能手机这几年发展迅速，移动应用程序的开发也已经俨然成为一个主流的分支，许多开发人员都投身到移动应用程序开发领域之中。如果一个.NET背景的开发人员，想要进入到移动开发领域，看起来似乎唯一的选择就是微软的Windows Phone 7，然后使用Silverlight或XNA开发Windows Phone应用。但现在人人都知道，当今最流行的移动平台非iPhone和Android莫属。由于.NET平台天生的特性，没人能指望微软能为iPhone或Android设备提供定制版本的.NET运行时。

不过幸好有了Mono，.NET开发人员仍然有机会尝试为iPhone或Android开发移动应用程序。Novell公司刚刚宣布了MonoDroid，一个将Mono与Android拉到一起的工具。MonoDroid将Mono运行时移植到了Android平台上，这样.NET开发人员就可以使用C#，基于MonoDroid来编写Android应用了。除了Mono运行时自带的Mono基础类库之外，MonoDroid还提供了针对Android的API接口。通过这些API，基于MonoDroid的程序就可以与Android系统和设备打交道了。安装了MonoDroid SDK之后，开发人员可以直接打开Visual Studio 2010，使用“MonoDroid Application”模板，创建一个MonoDroid项目，然后使用熟悉的C#开始编写代码。MonoDroid也会为MonoDevelop提供相应的支持，让开发人员使用MonoDevelop创建MonoDroid项目。

通过Mono，.NET开发人员还可以将自己的技能延伸到iPhone和Android平台之上。让我们想象一个有趣的场景，如果一个.NET团队需要创建同一个移动程序的Windows Phone 7版本、iPhone版本和Android版本，如果设计得当，将涉及各个平台不同的展现层与逻辑层分开，那么实际上开发团队可以很容易地在不同版本的程序中，共享大量的代码，节省许多的开发时间。

随着Mono不断地为.NET技术社区提供越来越多的工具和选择，笔者相信Mono的前景应该是越来越好。如果你也希望让自己的程序能运行在Linux、Mac OS X、iPhone、Android上，别忘了检查一下mono-project.com。

本月热点

● 会议

Microsoft HTML5 Labs

HTML5 Labs是微软发布的一个网站，专门用来展现微软所实现的HTML5中尚未标准化的各种技术原型。现在在HTML5 Labs上包含了IndexedDB和WebSockets这两个原型，前者是一个用来在浏览器中存储结构化数据的轻量级数据库，后者是一个用来实现浏览器到服务器间非HTTP式通信的协议。HTML5 Labs的网址是http://html5labs.interoperabilitybridges.com。

ASP.NET Web Platform Firestarter

ASP.NET Web Platform Firestarter是一组包含了6个课程的技术讲座，它讲述了ASP.NET 4平台和Visual Studio 2010提供给Web开发人员的各种新功能，包括ASP.NET 4.0 WebForms、ASP.NET MVC、WebMatrix等内容。

视频网址：http://channel9.msdn.com/posts/ASPNET-Web-Platform-Firestarter-Part-1-of-6-To-the-Web-with-ASPNET-4-Web-Forms。

CES 2011 国际消费电子展

虽然CES 2011并不是微软的会议，但在CES 2011上，微软宣布了有关下一个版本Windows的一则重大消息，那就是下个版本的Windows将可以运行在System on a Chip (SoC)架构之上，并增加了对ARM的支持。微软在CES 2011现场演示了运行在高通Snapdragon芯片和NVidia的Tegra芯片之上的Windows。有关这则消息的更详细信息请查看http://www.microsoft.com/Presspass/Features/2011/jan11/01-05SinofskySOC.mspx。

主持人：
钱宏武，现任职盛大创新院，原搜狐互动产品开发部主管，资深互联网社区架构师，垂直搜索领域专家。

互联网：新年的总结与展望

Web

本月热点

事件

2010年十大危险IT技术 苹果操作系统上榜

最危险的操作系统：苹果的Mac OS X；最危险的顶级域名：CO.、CC.。

评点：啥东西都有缺陷的一面，老外总结也一样。

2011年五大网络前端技术展望

- jQuery Mobile、jQuery 移动应用；
- 浏览器硬件加速技术；
- Node.js技术以及服务端JavaScript技术；
- 点击分享实时化；
- NoSQL Database，无需SQL语句的数据库技术。

评点：对于新的技术应用来讲，JavaScript会大放异彩，会的朋友应该继续加深，不会的建议去好好学学吧。

微软发布开源CMS平台1.0正式版

微软发布了名为“Orchard”的开源内容管理系统（CMS）1.0版。它是微软“Oxite”开源CMS系统的继承者，2010年12月，微软将Orchard从自己的CodePlex迁移到 Outercurve基金会，并承诺提供三年的技术支持。Orchard旨在帮助用户创建和管理网站，它和ASP.NET MVC3应用程序相似，使用Razor视图模板和SQL CE4来实现数据存储。

Web服务器：Cherokee 1.0.16 发布

Cherokee号称是目前最快的Web服务器软件，在性能上，甚至比nginx还略胜一筹。

Cherokee的功能包括支持FastCGI、SCGI、PHP、CGI、TLS及SSL加密连接，虚拟主机，授权认证，实时编码，载入均衡，与Apache兼容的log文件等。

年关时候：总结、奖金、年会总少不了，让我们从不同维度来总结2010年的互联网。

有人的地方就有是非，即使号称高素质、高科技的互联网也是如此。中国拥有数万家互联网企业，但矛盾最大、争议最多的，却就是那么几家：360、腾讯、百度、金山等。所以，如果列出2010年互联网十大争端，基本是这四家的排列组合。

对于互联网这个以技术起家的行业，初期是你做你的、我做我的，互不侵犯、后来则是有所磕碰，但基本都是很快和解。但随着现在互联网新用户的增长不再迅速，彼此的小摩擦终于发展出大阵仗：你限制我的软件，我抢注你的网址，你攻击我的服务器，我拔你的网线……技术就像一把刀，最初是披荆斩棘的工具，发展到今天却成了损人嫁祸的必备武器。

任何东西都有两面，取决于是用来救人还是伤人。2010年显示这种伤人趋势慢慢在壮大，希望它有一天能得到一定的遏制，毕竟这个市场才刚刚开始，每个领域都有无穷的想象空间和利润。而这种小家子气的内讧，实在不利于自己和整个行业的成长。

但是和为了吸引眼球、博君一笑的总结不同，预测才是我们真正需要关注，就像工作总结领导未必看，工作计划才是最关键的。

首先上榜的应该是云计算，每年的预测基本都会加上一条“云计算的普及”，今年也不例外。Amazon的成功让无数商家认为就在当下了，但真正进入后，你会发现这个市场远没有想象得那么大，特别在国内，相关法律、法规和商业环境的缺乏，让很多的企业和商家顾虑多多。在很长时间里，云计算的普及，都将仅仅是预测的榜上常客。

移动支付很早就开始说了，移动钱包、手机支付，更是早早就开始喧嚷，但除了一堆恶意扣费的ICP，移动支付并没有真正普及开来，特别是国内。移动支付之所以一直做得不温不火，缘于所有东西都是灰色地带，而且移动支付只是移动、联通们在做而已，但它们却拥有国内80%以上的市场份额。除非它们真正能够理解开放，和其他的商家共同开发，并且将合理的利润分给商家，否则移动支付的前景也只能是镜花水月，永远驻留在预测榜单中。特别是中移动，初期能把自己的渠道定位做好，就已经非常不错了。移动支付市场极大，如果还想一切通吃，实在是逆天了。

真正有意思的预测是地域相关的互联网服务，如照片、SNS、地图、问路……对互联网应用来说，一向是标榜跨地域的，现在终于返璞归真了。毕竟虚拟生活再精彩，我们还是要吃、住、玩，这些除去地域则没有任何意义。经典物理告诉我们，只有时间和空间是不可压缩的。面对这些不可压缩、无法处理的东西，用一个技术的手段来管理，这里面的商业价值不言而喻。所以这个预测，应该代表了今年的趋势。谁能抓住，就看不同公司的眼光和手段了。

浅谈次时代动作网游开发

主持人：
宋忆疆，2000年开始从事游戏程序开发，参与的项目《碧血情天》，《傲世三国2》，《乱舞天下》，《流星蝴蝶剑OL》等的研发，目前担任《流星蝴蝶剑OL》项目制作人。

游戏开发

MMORPG经过10多年的发展，已经进入平稳发展阶段，技术成熟了，模式也成熟了。

于是，最近两年，一个新兴的游戏名词“MMOARPG”，频繁地出现在大家面前。别看就多了一个代表动作的A字，但对于网络游戏来说，它很可能是一场风暴。而在17173新游期待榜中，排名前面的游戏更几乎都是动作类网游!

然而动作类网游，还是存在着几个问题，亟待我们去解决。

一、动作游戏和RPG交互操作的统一协调

动作类网络游戏，要怎么操作才能更方便？很多游戏机上的操作设计非常优秀，但是在MMORPG这个框架下，又会有一些问题，比如目标选定，在MMORPG中，很多界面非常复杂，需要用鼠标做大量交互操作，要选择目标等等。在战斗中，是否有选择目标，这个也让我们纠结了很久，有的玩家希望没有目标选择，随意发动招式，可以很自由的控制；而有的玩家希望自己更多精力是在做战术的抉择，而不是类似角色朝向这样具体的微操。最后，我们作出一个妥协，提供更好的Table选择目标对象的方法，方便辅助战斗锁定，但是也提供玩家关闭辅助锁定的选项，希望能满足更多玩家的需求。其实我个人觉得，3D视角下，要自己控制太多操作，真的很累。

二、网络延迟带来的同步困扰

网络延迟是要求高同步的动作类游戏的最大天敌。制定同步策略时需要注意以下几个原则。

- 位置同步是一切同步的基础
- 合理的客户端，服务器状态预测是保证流畅的法宝
- 找到合适的强制同步点
- 利用合理的状态消化同步差异

三、连招控制与反控制的平衡

动作游戏，连招是一个很爽快地玩法，很多单机动作，格斗游戏，甚至都有一些无限连的套路。但是在MMOARPG里面，这个就不能完全这样设计了，首先网络环境就不会允许，其次无限被连的人的情绪我们也必须照顾到。所以游戏的控制与反控制设计，是游戏平衡的最根本。

四、高精度高效率的客户端服务器碰撞检测

惊险的关卡设计，诡异的机关，飘逸的轻功，让动作类游戏的体验感直线提升。但是这些的操作，都和碰撞检测休戚相关。高精度的碰撞检测，往往意味着执行效率的降低，而服务器的处理，更是让计算机承担很大的压力。这个部分，开发人员的优化要不遗余力。客户端部分，我们可以采用更加精确的碰撞体，而服务器端在保证足够的需求下，要尽可能减少复杂度。当然采用什么样的碰撞检测算法，也很重要。

次时代动作网游大潮正滚滚而来，作为游戏开发者，将来应该怎么制作出更加优秀的MMOARPG呢？希望更多的从业者、玩家去思考和关注这类游戏。

本月热点

新品发布

动作网游《Tera》即将韩国公测

BlueHole工作室动作类网游tera很快就要面对韩国玩家，经过多次测试后，工作室调整了大量游戏内容和平衡性。Tera在公测，等级封顶在38级。玩家在满级后，可以体验到游戏的政治系统，并参与领主选举等。

中国明确网络侵权标准

最高检、最高法将在近期公布《打击知识产权刑事犯罪适用法律若干意见》，这将为打击网络侵权案件提供重要的法律依据。游戏“外挂”侵权案定罪、量刑标准等都有望推出明确规定。

大华建设收纳斯达克退市通知

唐骏旗下的联游网络借壳后，大华建设曾经短时间股价站在1美元以上，但是公司业绩没有根本好转。由于大华建设普通股的最低买价已连续30个交易日低于每股1.00美元，该公司收到普通股将从纳斯达克全球精选市场退市的通知，大华建设拟提出上诉。

掌上明珠推出国内首款Q版神话题材的手机网游

《明珠西游》。该游戏由非鱼工作室倾力研发，风格是2.5D，采用了明珠西游自主研发的引擎，首创了手机网游中空战、海战、地府战系统，对回合制战进行创新，开发出神魔双修，提高了游戏的自由度和个性化。可以说，在移动互联网应用日益繁荣的今天，《明珠西游》也称得上一面民族的旗帜。

主持人：
马宁，微软最有价值专家，Windows Mobile开发者。

2011年关键词

新平台

本月热点

新品发布

Apple Mac App Store

1月7日，携iPhone App Store大胜的余威，Apple Mac App Store上线。App Store改变了软件分销模式，从而从根本上改变了软件市场由几家大公司把持的格局，所以，在iPhone的世界里，有那么多创造奇迹的小公司。Mac App Store目前有几千款软件，虽然无法撼动整个PC市场的格局，但如果微软继续无视这种新的营销模式，很难说PC领域不会出现革命性的App Store。

可弯曲屏幕

1月8日，三星发布4.5寸可弯曲屏幕，AMOLED最大的卖点就在于其可弯曲的特性，三星这次推出的可弯曲屏幕厚度只有0.3mm，可以用手指任意弯曲。尽管距离量产还有一段距离，而且电子报纸也并不实用，不过相信我们以后会看到越来越多异型的电子设备了，比如圆形的水晶球？

Angry Birds多平台版本

1月7日，Angry Birds 推出 PlayStation 与 PSP 版本，在推出iPhone、Android和Symbian版本后，愤怒的小鸟正式发布PlayStation和PSP版本，这也是愤怒的小鸟第一次从智能手机平台迁移到家用游戏机、游戏掌机平台。除了分辨率之外，最大的挑战来自于输入方式的改变，从触屏游戏改成通过手柄输入，相信游戏开发者要有很多问题处理。

2011年1月6日，一年一度的CES在拉斯维加斯举行，作为每年第一场世界级的消费类电子展会，CES具有产业风向标的意义。今年的几个关键词是“Windows、ARM、3D、体感”，让我们逐个进行剖析。

Windows，熟悉IT历史的人恐怕对Wintel这个词不陌生，这个辉煌无比的联盟如今却被雨打风吹了：Intel在忙自己的MeeGo，而微软的Windows 8则宣布支持ARM。

其实这也不奇怪，x86近几年没有太大变化，而ARM却从移动设备逐渐向平板、上网本、服务器等领域不断扩张，其运算能力已经完全可以支持Windows的运行了。

与时俱进是正确的，不过Windows 8还有很长的路要走，比如驱动程序和应用程序的兼容性、针对不同设备类型的支持等，这些都是Windows 8需要解决的问题。

Windows 8未来针对的领域可能有两个：运算能力较强的移动设备和节能型的服务器集群。

ARM，无疑是今年CES上最闪耀的一个单词。在ARM中，Tegar 2则是璀璨星空中的北极星：新Moto发布的首款Android 3.0平板Xoom，采用的便是Tegar 2；Acer支持LTE的Iconia平板用的也是Tegar 2；而国产天语W700用的也是Tegar 2；甚至连车载电子领域的Tesla Model S也配备了基于Tegar的中控电脑，据说现在BMW也在与NVIDIA接洽。

电子产品研发周期一般是一年时间，Tegar 2是一年前性能最为优秀的ARM芯片，其出色的3D加速性能是最大的卖点。不过由于NVIDIA公开技术资料较少、技术支持团队规模有限等限制，估计Tegar很难像MTK那样进入山寨机市场。

3D，是今年CES的另一个热点，大屏幕3D电视、移动3D设备会成为接下一年的发展重点。裸眼3D和眼镜3D技术是目前两种主流的3D技术，这两者都有一些技术问题有待解决：裸眼3D对于观看角度有严格限制，很难同时供多人观看，而眼镜3D技术则需要佩戴眼镜。

所以，我们目前看到的技术还都不够完善，比如东芝推出的裸眼3D电视，支持4K刷新频率，但观看区域受到严格限制。目前，三星和ReadD推出的RDZ 3D眼镜技术，使用了较便宜的偏光眼镜，在电视机上加装额外的液晶面板，实现两个偏光方向的快速切换。

此外，三星还推出了大屏幕的LED背光3D电视和全球最轻的3D眼镜，看来三星准备往时尚方向发展了。Sony的解决方案最为梦幻，采用了3D头戴式显示器，不过目前还只能用双手举着。

最后的关键词无疑是Kinect了，有人评价Kinect是可以达到“神器”级的电子产品。而微软在CES上推出了Avatar Kinect，支持玩家的化身出现在游戏之中。未来Kinect还会给玩家带来怎样的惊喜？看来在体感方面，我们只是刚开始而已。

2011，Pad元年

主持人：
崔海斌，十几年的开发经验，专注于Java、Android、移动开发等领域。现就职于创新工场旗下的点心团队就任总架构师。

移动

2011年开年大戏CES2011在1月6日到9日上演了，在这次大展上，最大的赢家就是Nvidia和Motorola，两者的成功主要倚靠的都是Android，尤其是Motorola，全部设备都基于Android。

Nvidia凭借其双核Tegra2终于赶上了移动设备的大潮，之前总是觉得Nvidia总是雷声大雨点小，在各个移动平台都是行动缓慢，这次一举杀入智能手机和Pad平台，相信其未来的发展一定会进入新的阶段。

而Motorola在CES2011之前刚刚完成了公司分拆的任务，新的Moto移动带着多款手机和Pad产品参加了这次CES2011的展会，而媒体和参展者也给予了Moto最好的回报，在CNET的展会最佳产品评选上，Moto的Pad产品Xoom和手机产品Atrix 4G分别当选Pad类和智能手机类产品的最佳，而Xoom更是当选全部产品的最佳。

而之前出现的所有Android 2.2甚至更早版本的Pad均在其面前黯然失色。从Honeycomb的演示来看，其不愧可称为iPad的真正对手：全新设计的UI、全新设计的应用、全新的操作方式和屏幕布局，甚至还有视频通话，而iPad的视频通话还要等待iPad2才可能推出。不过Xoom不是唯一一款搭载Honeycomb的Pad，在展会期间放出视频的还有来自T-mobile和LG共同准备推出的G-slate，相信会给最近情况不佳的LG注入一剂强心针。

有消息称Xoom会在2月内发布。而在Honeycomb的发布后一定会一统基于Android的Pad，如果不能升级到Honeycomb的Pad发展估计不会太好。不过此次展会上展出的Pad产品有几十款，大部分产品都是搭载早期的Android系统，还有基于Windows 7的Pad，再加上RIM的Playbook和HP的webOS Pad，由此可以想象2011年将会是一个Pad产品百花齐放的一年。谁能动摇iPad的地位？从目前的形势来看，Android存在最大的可能。但随着iPad2在将今年第一季度推出，相信2011年，iPad在Pad方面的头把交椅应该不会失去。

Apple虽然缺席了CES2011，但其新闻也是不断。新年伊始，所有iOS用户收到的第一个礼物设备居然是闹钟无法正常响起的Bug，不过此Bug并未对Apple造成太多困扰，Apple的最新市值突破了3000亿美元大关，超越中石油，成为全球第二大市值公司。而其CDMA版的iPhone4于1月11日与Verizon共同发布，相信这才是市场对其利好的反馈。

同时，近日关于iPad2的小道消息也是不断，有传言说2月就会发布iPad的新版本，不过从目前的情况推测，iPad2除了分辨率的提升以及前后摄像头以支持Facetime以外应该不会有太大的更新。

RIM的Playbook最近由于耗电问题的困扰，产品迟迟无法发布，而其手机市场份额进一步缩小。不过也有一个利好消息，国内的联通也准备定制一款黑莓手机，将于一季度上市。

本月热点

● 事件

魅族M9正式发售

2011年1月1日，魅族的M9正式发售，据报道，首日发售场面相当火爆，M9大受魅族粉丝的追捧。魅族长时间沉寂的力作，其性价比还是相当不错的，希望其作为国产手机中的一支优秀团队越走越好。

Android成人气最高手机操作系统

据尼尔森2011年1月公布的11月份美国市场数据显示，在过去6个月时间里购买智能手机的用户中，购买使用Android操作系统的智能手机的用户所占比例为40%，使其成为最近购机者中人气最高的一种操作系统。从整体市场份额来看，Android操作系统所占份额为25.8%，仍落后于苹果iOS的28.6%。与此相比，RIM黑莓所占份额为26.1%，处于苹果iOS和Android操作系统之间。看来Android手机冲入美国智能市场第一应该不会很久了。

飞思卡尔下一代处理器 i.MX6

芯片制造商飞思卡尔半导体发布了自己的下一代处理器 i.MX6，四核的Cortex A9架构。移动设备已经进入双核时代，看来四核也不远了，下一步最需要革命性的发明就是电池了。

主持人：

浦石明（花名法海），目前任职于淘宝网UED前端开发工程师，专注于Web表现层相关技术，对各种应用、开发工具也有广泛研究。

对 JavaScript 应用编译技术

Web前端

本月热点

● 新品发布

Firebug

2010年11月29日，Firebug1.6发布。Firebug是前端开发人员的必备调试工具，距离1.5版本的发布已经相隔很长时间，本次更新有很多令人瞩目的新特性，值得一试。

Opera11

2010年12月16日，Opera11发布。虽然市场份额不大，但是Opera极富个性，并且遵守规范、性能很好。与上一代相比，新推出的Opera11体积减少了30%，而性能提升了30%。

● 事件

HTML5实验室

微软推出HTML5实验室，可体验HTML5新技术。HTML5是尚在草案中的新规范，微软为了推动标准也是下了十足力气，包括推广IE9等。HTML5时代将要到来，可先于此实验室体验一把。

● 会议技术

Velocity大会

2010年12月7日，Velocity大会在北京举办。此次大会是国内外互联网公司为提高网站性能而举办的技术交流大会。在此大会上，包括Google、FaceBook、淘宝、腾讯等国内外著名公司都会有精彩的分享。

D2前端技术论坛

2010年12月18日，第五届D2前端技术论坛在杭州成功举办。D2论坛是前端业界的一个重要的交流平台，D2的举办是业界精英的一次聚会，可以加强业界人员对技术的讨论和对前端发展的展望。

JavaScript（下简称JS）是一种解释型的语言，这意味着，它不能像编译型语言那样能在编译期得到检查和优化。这在很多方面产生了一些负面影响，比如，不能在执行前规避语法错误、代码得不到优化等。随着Web应用复杂度的加深，网页对JS的依赖越来越严重，应用中的脚本越来越庞大和复杂，同时由于JS的动态性和弱类型等特征，脚本易出错、难调试、性能低等问题日益凸显。

追溯到2009年底，Google向业界开放了它的Closure Tools，这是一套面向前端开发的工具集，其中包括了针对JS的编译器——Closure Compiler，它为JS的编译提供了可能性。另外，在2010年12月的D2大会上，有一个议题正是《打造高品质的JavaScript——Closure Compiler》，讲者介绍了使用此编译器的方法，以及如何通过一定的编码约束来达到对JS代码的检查和优化。

Closure Compiler的工作包括：检查源代码中语法、语义、语用上的错误，压缩和优化源代码。此前，也有如YUICompressor这样对JS代码压缩处理的工具，但是和Closure Compiler并不属于同类产品，需要注意的是，Closure Compiler做了更多工作。而Closure Compiler不同于C等语言编译器的地方在于，它处理的对象是JS，不需要产生其他中间代码或汇编代码/机器码，因此输出的还是JS，但是是经过分析的、优化后的JS；另外，它也可以选择输出“Parse Tree”（使用--print_tree参数），所以，它的确完成了一个编译器需要实现的功能。

Closure Compiler包含了三个编译级别。级别越高，则优化效果更甚，同时对开发者的约束也更大。要真正做到对JS的编译，必须开启“高级模式”。除了移除空白和注释之外，它提供了以下的额外优化：

- 更激进的重命名，如obj.property改为a.b，将深度过高的命名空间平坦化等；
- 移除垃圾代码，如删除未被调用的方法定义，警告逻辑死角；
- 将函数内联。

要达到高级模式的预期优化效果，开发者必须对自己做一些约束，因为JS是弱类型、动态性的，否则JS的这种灵活性将使编译器无能为力。总体上，这种约束包括限定某些JS编码风格，以及使用相应的JSDoc注释。限定的编码风格主要是强制开发者以强类型的静态语言风格编写JS，将关注点从运行时的动态技巧转移到组织代码、编写逻辑本身，而可能由弱类型系统和动态特征产生的问题和风险则交给编译器。

总之，使用Closure Compiler对JS进行编译，将对代码进行大幅度的优化，同时，编译前的代码也将变得高一致性和易读，虽然失去了一些JS的灵活性，但是这对于大型开发团队或构建中大型Web应用是非常有帮助的。通过Closure Compiler，我们可以想象，未来对JS进行编译的工具应用，将变得越来越广泛。

整合与碰撞

主持人：
肖新光，网名江海客，安天实验室首席技术架构师，研究方向为反病毒和计算机犯罪取证等。

安全

启明星辰在元旦停牌之后，宣布的收购对象是联想网御，这个整合让一些业内人士连呼意外。从可以获得的销售数据来看，尽管启明星辰是以旁路产品IDS起家，但其启明天清汉马系的网关产品的出货量已经大于了联想。从联想网御的产品线路来看，除了VPN和加密产品外，对启明星辰的已经比较完备的产品线没有更多的互补性。其收购可能更多考虑的是市场互补的因素，如行业重心、渠道等因素。

这是启明星辰在上市之后，第一次让业内感觉到资本的力量，但我们也看到了其进取的雄心。固然这个收购的量级，与去年下半年IBM、Intel、HP一串惊天动地的大并购相比，显得微不足道，但对羸弱的中国信息安全产业基础来说，这种企业级产品企业之间的资本整合远比桌面市场的山寨、克隆与恶战良性。

而从资本角度来看，其实更值得关注的是网秦接受HTC等战略投资者的投资。不管业内对网秦的技术底蕴如何看待，其上市前的资源储备可谓充沛得多。基本来看，今年网秦上市几乎已经是必然。网秦上市后，究竟会在这个规模并未成熟的市场如何展开，需要进一步观察，但可以肯定手机安全大有作为。

而桌面本月再爆隐私之争，其起源可能是由于360上载服务器的配置错误，从而使客户端上载的URL访问记录文件可以被Google的Spider 抓到。而在Web为王的时代，当用户名和密码都成为了URL的一段参数，URL访问记录的安全价值显然不言而喻。由于在这些记录中能找到一些邮件用户名和密码，甚至网站管理系统的登录链接，这个问题成了Google Hacker们的新话题。而就在此时，金山召开了发布会点破此事。而360则解释上载URL是为了进行有害站点发现和过滤。360在1月3日进行了舆论反击，指认金山旗下站点金山云安全中心也可以查到类似包含用户名和密码的URL。金山的应答是，所谓类似问题的URL是从1日起，有人陆续通过恶意网址举报功能添加进来的，而不是由金山的客户端上报的，而360的客户端不但进行完整的URL访问记录上报，还做了用户唯一标识码。

但此时媒体和公众似乎在3Q大战后，失去了热情，这本该更值得业内警醒的事件，并未登上头版。但面对网那边那朵巨大的云，也许这并不只是一个技术规范的问题。

但是，“狼来了”的警告还是需要被认真对待的。国家信息安全漏洞共享平台（CNVD）发布了第五十一期漏洞通报，CNVD其将本期信息安全漏洞威胁整体评价级别为低。本期共收集、整理信息安全漏洞52个，其中高危漏洞4个、中危漏洞15个、低危漏洞33个。上述漏洞中，可利用来实施远程攻击的漏洞有45个。CNVD宣布，网上已经出现针对“Microsoft Windows Fax封面编辑器远程代码执行漏洞”等零日攻击代码。详情请见：http://www.cnvd.org.cn/。

本月热点

● 资讯

发现Chrome浏览器漏洞获Google重奖

Google向一位名叫Sergey Glazunov的开发人员颁发首个“精英”Chromium安全奖，金额达3133.7美元。他的最大发现是Chrome浏览器“对话控制中的一个指针”存在重大安全隐患。此外，他还发现了其他4个“高危”漏洞。Google此前决定，向那些发现Chrome浏览器重要漏洞的开发人员发放数额不菲的现金奖励。

IE的HTML渲染引擎中存在0day漏洞

微软证实日前曝光的一个IE漏洞可能会导致远程代码攻击。这是一家法国安全公司Vupen曝光的0day漏洞，微软随后开展了调查，就在Vupen公开了攻击代码的同时，微软也发布了安全公告，警告用户避免受到此漏洞的影响。

亚马逊云计算服务可帮助黑客攻破WiFi网络

德国科隆的一位计算机安全顾问Thomas Roth称，他能够使用一种特别制作的软件攻破采取保护措施的网络。他编写的这个软件在亚马逊基于云计算的计算机上运行。这个软件利用亚马逊的高速计算机每秒可测试40万个潜在的口令。

欧盟称公共云计算使用有风险

据欧洲网络和信息安全局（Enisa）最新发表的报告称，私有云计算环境比公共云计算更适合政府机构的需求，尽管公共云计算能够提供更高的服务可用性和改善的成本好处。这个观点看，云计算是风险最大的，因为拥有云计算服务的公司可能是非欧盟的公司，对于隐私、恢复措施和潜在的突破不能提供充分的透明度。

CES 2011——少数派报告

■ 文 / 张辉

CES是消费电子展的缩写，是全球规模最大的消费类科技产品交易会之一。2011年CES正式展会从1月6日~1月9日，共四天。几天参展下来，发现真正在CES 2011首发的重头产品并不多。大品牌厂商现在往往更倾向于在自己独立的环境里，用自己喜欢的方式发布自己的重磅产品。

即使这样，第一次亲身接触CES，能够亲眼目睹和切身体验到一些令人印象深刻的产品和技术，不是一句“物有所值”所能概括的。下面逐一揭示那些给我印象最深的项目。

三星Smart TV沿用3D+Smart的设计思路

在本届展会，大家不约而同地都会提到四块屏幕的问题，这就是TV、平板电脑、手机以及PC。相比以往，这四块屏幕之间的界限与分工会更加清楚，而这四块屏幕之间的无线连接和内容共享也更加方便。

Smart TV是一大热点

三星和LG在会展中心附近的大楼上布下巨幅广告，阐述自己的Smart TV概念，Sony也在中央大厅的入口上方有面积稍小的广告，介绍其基于Google TV的联网电视产品。LG和三星分别占据会展大厅的核心的位置的大片展区，在面积、气势、展示方面几乎不相上下。相关展出的主要的思路都是3D＋Smart，也就是立体影像外加智能电视，同时，借助家庭的无线网络，把网络存储（NAS）、手机、平板电脑等无线连接起来。

平板设备各擅胜场

此次参展的平板电脑很多，但给人留下印象深刻的不多。选择CES首发的Motorola Xoom是第一款基于Android Honeycomb版本的平板产品。虽然Motorola低调地把该平板放在一堆Android手机后面的角落里，但是依然引来众多参观者。Xoom还是保持了一些神秘感，任何参观者不能触碰Xoom。Motorola的员工利用Xoom硬件播放事先录制的视频，演示了Honeycomb的基本特性。其他两个体验不错的平板设备都是来自于NVidia的Tegra2展示样机，其中一个是Dell的Streak 7。在玩视频游戏时，流畅度和画面效果可以让你忘记iPad。

体验是最关键的问题。良好的体验有赖于硬件和软件两方面的成熟度和整合度。在这一点上，采取高端解决方案的厂商产品具有一定优势，但这样会导致Android平板与iPad之间的价格优势减少很多。

虽然PlayBook并非此次CES才宣布，但在本届CES上，大家可以试用PlayBook。PlayBook的选材和做工上乘，触摸感觉和操作流畅度让人想起iPad的体验。而在多任务方面，PlayBook更是胜出一筹，用手势控制多任务的切换以及程序关闭、用手势控制程序Dock等都令人眼前一亮，让人有WebOS的体验感觉。

软件方面，Honeycomb是Android转为平板设计的第一代产品，对其用户界面、用户体验的成熟度还需要观察。

还有比较重要的是，针对平板电脑的Android Market目前还没有看到消息，大家面临的问题还是缺少典型的平板应用。虽然Android之前的版本就支持多种分辨率、多种DPI，但是对于软件而言，屏幕尺寸也是一个问题。在这一点上，专为iPad设计的一些软件比如Flipboard就充分说明了这一点。除了Google的Team专门为Android的Honeycomb版本重写的Gmail、Maps、YouTube和Gtalk等Google自有的软件之外，还缺少第三方程序。问题是，除了

有家电大厂会出钱外包定制之外，有哪些程序员会花更多工夫给设备数量更少、硬件规格更不同一、盈利模式尚不明朗的Android平板专门开发程序呢？

Android手机前途无量

Android手机发展历史是一个快速发展、快速迭代的历史。典型的案例就是国内Android玩家对于HTC手机的简称，从G1、G2一直到现在G8等。而Android软件的版本，也一路从Cupcake、Donut上升到现在的Froyo和Honeycomb。这种快速的演进、不断的Beta测试、不断的升级，正好是互联网的速度和习惯。但不同于互联网的一点是，客户必须通过更换手机硬件才能体验到最新的软件。

手机不同于其他家电之处在于被认为是可以随身携带、随时拿出来的时尚产品，所以夸张的说法是，国外用户的换机频率已经到了六个月一部。所以，这种节奏和Android软硬件升级换代基本上是匹配的。

Android在手机上绚丽的发展才刚刚开始，但对于电视、机顶盒以及平板等其他产品的进军过程，还是值得去等待和观察的。

4G和社交网络

不仅在会场内，各大厂商争奇斗异。在会场外，他们也进行着广告比拼。Sprint做的广告以4G和Android为主题，遍布机场、酒店外立面、穿梭巴士。HTC的广告“4G，for the people”也令人印象深刻。这种铺天盖地的4G广告，让每个与会者都感到“不远了”，起码在北美。

在国内，大家总觉得Facebook、Twitter之类的估值太高。但是到了拉斯维加斯，到了CES，这一切基本上就不再是疑问了。酒店、商场、品牌产品的广告甚至是小商店门口的招贴广告都有自己的Facebook和Twitter信息。当传统广告主比以往更加重视社交网络，而这一切又看起来更为时尚的时候，社交网络在北美已经在互联网之外体现自己的价值。期待社交网络的中国故事。

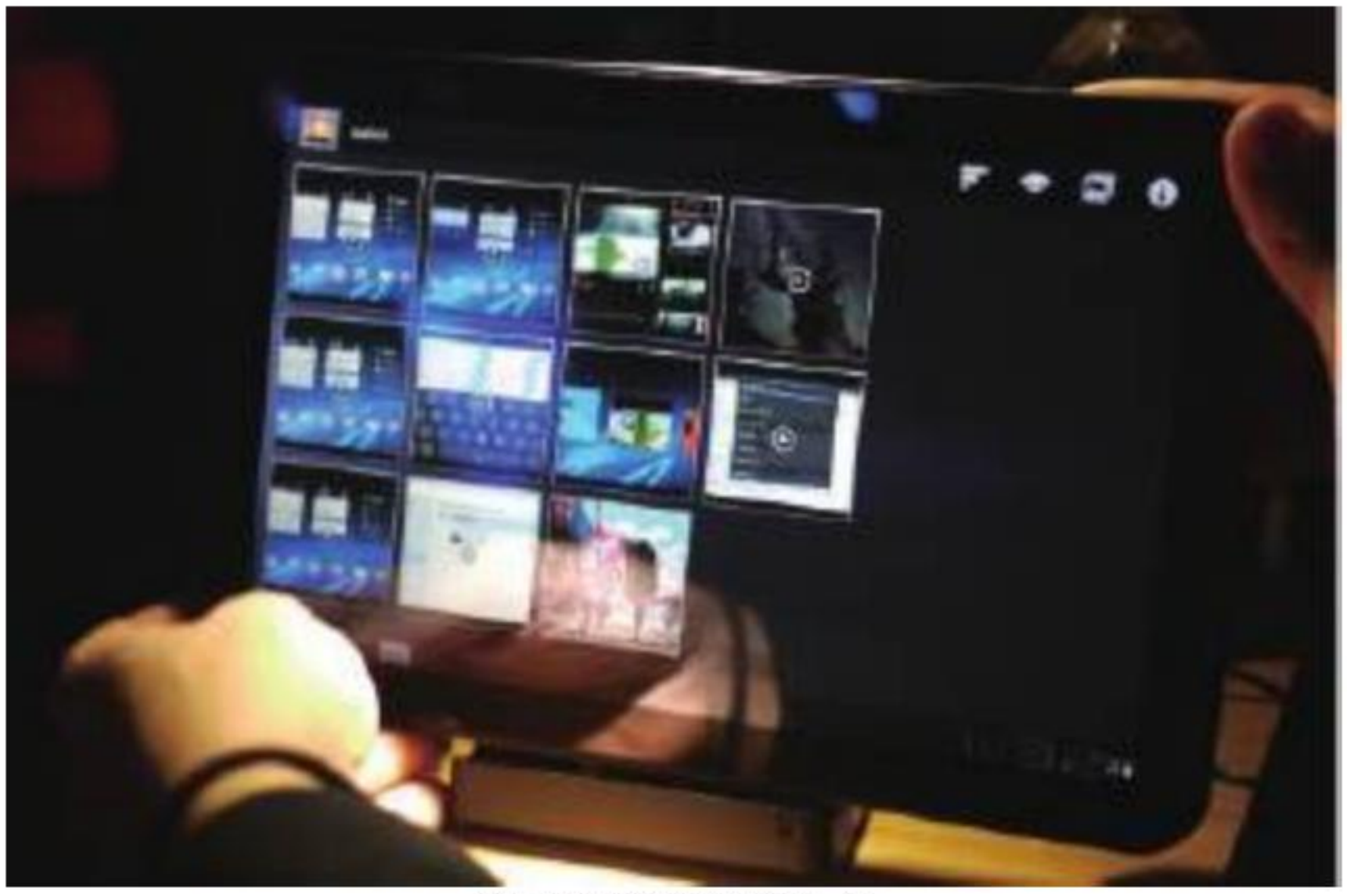
对软件开发人员的建议

应用成为体现设备价值的核心。未来的智能设备从PC、手机扩展到平板和电视，应用成为体现设备价值的核心。而由于这两方面相对于PC和手机而言都是增量的市场，所以，对于优秀软件开发人员的需求会比以往更加强烈，市场价值会更高。对于优秀开发人员的巨大需求和此类人才的供应不足会成为突出矛盾。在国外那种核心编程人员的薪资和VP一致的情况会不会在中国出现？我认为在智能设备领域很可能出现。所以希望大家能继续关注这一领域，坚持学习和实践相关技术。

智能设备的产业链日趋成熟。CES上有很多中国台湾地区的厂商，并没有展位，而是在附近的酒店租了专门的房间，与相关渠道和品牌客户进行商务洽谈。他们专注于做核心技术和解决方案。然后把解决方案授权给有品牌、有渠道的厂商，由他们卖给最终消费者。方案商和品牌商各司其职。

在智能设备软件开发领域，产业链也会逐渐成熟起来，这就像硬件的制造和销售，或者电影的制作和发行。在未来，需求、创意、研发、设计、营销、渠道等各个环节会充分成熟起来。

大型公司可以在各个环节四面出击，完整地做起来，而小型团队要根据团队特点，正确定位，充分发挥自己优点，或者做设计、或者做产品研发、或者做需求与创意，积极利用日益成熟的产业链和分工优势，来获得自己的成功。



Xoom的很多操作让人眼前一亮

小店门口的招贴广告，社交网络在北美很普及

黑莓实力

——BlackBerry Devcon Asia 2011印象

■ 文 / 刘江

2011年1月13日-14日，黑莓亚洲开发者大会在印尼巴厘岛举行。这是RIM公司首次在亚洲举行主要面向个人消费者应用开发者的技术大会。

为什么选择亚洲，选择印尼？在第一天的主题演讲中，RIM公司东南亚地区总裁Gregory Wade给出了答案：在印尼、马来西亚和泰国等东南亚市场，黑莓的风头已压过Android和iPhone。尤其是用户超过250万的印尼，是黑莓除美国之外的全球第二大市场。与此相对应，黑莓在亚洲开发者的增长速度也雄冠全球。在全球范围内，黑莓在超过175个国家拥有580多家合作伙伴和销售商、超过5500万订户。此外，全球最大的移动社会化网络BBM（3300万用户，即将向开发者开放平台），用户超过3500万、每日下载200万次的应用商店App World，灵活便捷的支付方式和完备的广告服务，共同组成了黑莓颇具潜力的开发大平台，吸引了30万以上的注册开发者。

PlayBook与Tablet OS

本次大会最受关注的当然是PlayBook。主题演讲中，RIM开发关系总监Mike Kirkup同时打开几个视频播放器、一个OpenGL渲染应用和一个3D游戏，还能流畅切换，充分体现PlayBook硬件和操作系统的强大。

从开发人员的角度来看，PlayBook的操作系统Tablet OS采用了老牌的类Unix微内核QNX Neutrino。QNX广泛用于车载系统、医疗设备、路由器等许多实时、关键任务的场景。微内核设计可以确保各组件故障重启时不影响其他组件和内核，保证很高的可靠性、容错性和可扩展性，此外设计本身已经很好地考虑了多核和多线程需求。由于表现出色，RIM官方之前已经确认未来的智能手机也将迁移到QNX上。

RIM宣布PlayBook将支持Adobe Flash/AIR、WebWorks、C++和Java四种开发方式。但本次大会上只发布了Tablet OS Adobe AIR SDK 0.9.2和开源Web框架BlackBerry WebWorks SDK。

其中，AIR SDK除了专门针对移动设备进行了性能优化之外，还增加了这些设备专有的API，如加速传感器、地理位置等。从SDK项目主管Julian Dolce的讲座来看，使用AIR进行PlayBook开发总体上非常方便，而且可以通过C++进行扩展以提供更好的性能。

而WebWorks是RIM以WebKit为核心打造的一个开源Web平台。SDK主要由命令行工具WebWorks Packager和JRE 1.6及文档组成，可以直接将写好的Web应用通过Packager编译，生成可以发布到PlayBook上的.bar应用文件。会议的讲座中，通过WebWorks开发应用，可以非常轻松同时部署到PlayBook和黑莓手机上。

值得一提的是，RIM还宣布，3月15日前，提交WebWorks应用的开发者有机会免费获得一台BlackBerry PlayBook平板电脑。相信已经有开发者怦然心动了吧。

The Web is Your App

Torch浏览器的创始人、现任RIM高级总监的George Staikos在主题演讲认为“The Web is Your App”。这是否代表了RIM公司的技术战略取向呢?

就此我专门采访了Staikos本人。他解释说，无论对最终用户还是开发者而言，真正有意义的是内容和体验，而不是呈现的技术方式。他相信现在的硬件发展正在消除Web技术普遍应用的障碍，Web技术能够跨应用和数据源集成，产生更丰富的体验，将成为跨设备和平台的最佳选择。RIM的PlayBook之所以会首先提供Adobe AIR和WebWorks作为开发平台，就是认为Web技术是目前应用开发的主流。此外，RIM还在会议期间发布了BlackBerry® Push Service SDK 1.0.1、BlackBerry 6.1 应用平台等新技术。

本次大会并不是黑莓技术的全面展示。事实上，RIM公司还有许多企业级和服务器端的技术，不少是非常领先甚至独门的。除了Devcon，RIM还有一个更大规模面向企业与合作伙伴并涵盖技术与商业的WES会议，2011年开始改名为BlackBerry World，2010年10月上海举行的“BlackBerry移动互联峰会”是这一活动的中国版本。

但是，这次大会还是充分体现了黑莓的技术实力和底蕴。相比其他更热门的平台，这里很可能蕴藏着更大的商业潜力，等待着开发者去挖掘。

搭建产品经理交流的专业平台

——记PMCAFF第一届产品经理峰会

■ 文 / 陆蔚青

2010年12月18日，以“开放分享、平等交流”为宗旨的PMCAFF第一届产品经理峰会在北京召开，吸引了来自全国各地近200名产品经理参加，活动围绕产品经理的专业技能、如何对待用户需求及产品人员的职业素养等问题展开了深入讨论，通过主题分享的方式促进了产品经理之间的沟通。

观众提问现场，火爆程度可见一斑

嘉宾分享质量高，专业性强

这次产品经理峰会上台分享的是几位颇具实力的产品人士，负责邀请嘉宾的腾讯产品经理霍莹表示：“原则是保证嘉宾既要来自产品工作的第一线，同时又要有相当的管理经验及高度。为此我们花费了很多心思来提名、沟通、甄选，目的就是让参与活动的产品经理们不虚此行，大有收获。”

著名产品经理读物《结网》作者、糗事百科网站创始人王坚分享的题目是《战略与执行——高级产品经理的基本素养》，从自己的互联网产品职业生涯开端讲起，融入在腾讯、飞信工作多年的心得体会，深入浅出，并举出多个产品实例，把产品经理进阶发展所需要的基本能力总结为战略理解、产品把控、市场分析、人际沟通、团队建设等，也强调了产品经理应该具有良好的文档撰写能力及具备积极分享的开放心态。分享结束后，他还送出了几本签名的著作。

深圳万兴软件有限公司副总裁谢宏中的分享题目是《产品经理负责制的诱惑与窘迫》，站在企业管理层的高度分析了不同的组织结构下产品经理角色定位，指出企业面前有理想中产品经理负责制带来的以业务为中心、专业化强这样的好处，也有现实中权责不一，可能走偏、危害长期利益的窘迫。明确了靠主业生存，产品间差异化明显并且独立核算的企业比较适合由产品经理负责一整个项目生命周期及营收的观点，整个分享过程更像是一堂优秀的MBA课程。

搜狗输入法之父、现360产品经理马占凯的分享题目是《猴子的香蕉》，这个分享是马占凯多年从事产品工作经验的浓缩。首先，通过为人熟知的浏览器、下载工具、播放器等产品之间的比较，阐明了需求是什么、需求从何而来、需求如何确定这几个产品经理时常困惑的问题，他指出真正的需求如同沙漠之水，是用户宁可付费也想要的东西。他还从用户需求客观存在，需要不同服务来满足的角度详述了为何在互联网巨头垄断市场的大环境中，仍然不断有像Facebook及Twitter这样的创新企业出现，也说明了为何看起来不起眼的小众市场同样可以诞生高市值的公司。

马占凯在分享中提出了一个公式：需求必要性=人数×次数×重要性，并用该公式构建了一个确定需求优先级的金字塔模型，还通过360浏览器、搜狗输入法的真实案例来证明了公式的有效性，内容生动有趣，台下反响热烈。

前联想365创始人之一、现阿里巴巴ASC资深经理李勇瑞的分享题目是《我眼中产品经理的基本素质》，从阿里巴巴的实际工作环境出发，说明产品经理是一个专注与好奇的综合体，需要具备一定的抗压能力及拥抱变化的胸怀，在有发展前景的业务平台上有所坚持、有所担当，就一定会有收获。

结语

近年来国内IT行业的同业交流氛围已经形成，尤其是在互联网公司扎堆的北京、上海、杭州等城市，比较著名的有以设计见长的UCDChina、以技术为主题的OpenParty、D2Forum等，现在又有了以产品经理为目标人群的PMCAFF。这些交流活动为IT从业者提供了向精英学习的机会，也是一个平等讨论的平台，是惠及业界的好事。

信息平台和数据科学家的兴起

■ 文 / Jeff Hammerbacher

Facebook有了“自知之明”

在2005年9月，Facebook首次向非大学生公开，允许高中生注册账号。忠实的用户愤怒了，但Facebook团队认为这是为网站做出的正常方向。那么它该如何证明它的方案是正确的呢？

此外，在几乎所有可登录Facebook网站的学校中，Facebook已经渗入学生当中，但还是在有部分学校中，该网站一直不受青睐。和那些更成功的网络相比，这些落后的网络对于Facebook有什么区别呢？Facebook团队应该如何做才能激励他们的成功？

当我在2006年2月参加Facebook面试时，他们正积极地期望找到这些问题的答案。我曾在大学学习数学，在华尔街工作近一年，工作内容是构建模型来预测利率、价格复杂的衍生产品和对冲抵押贷款池；有一定编程经验，GPA成绩“暗淡”。虽然我的背景可能不太理想，但是Facebook却给了我研究科学家的职位。

几乎同时，Facebook聘用了一位报告分析主管。该主管在解决问题方面的经验远远超过我。我们和另外一位工程师一起，开始着手构建一个数据收集和存储平台，以便找到我们产品以上问题的答案。

我们第一个尝试是构建一个离线信息库，其涉及两个方面：一是用Python脚本把查询分发到Facebook的MySQL服务器层，二是采用C++实现守护进程，实时地处理事件日志。当脚本可以如期运行，我们每天收集大约10GB的数据。我后来明白系统的这部分通常称为“ETL”过程，即抽取、转换和加载。

Python脚本和C++守护进程从Facebook的数据源系统中抽取数据，然后这些数据又被加载到MySQL数据库用于离线查询。我们在包含这些数据的MySQL上又运行了一些脚本和查询，对数据进行聚集，以便得到更有用的表现方式。这种用于决策支持的离线数据库即“数据仓库”。

最后，通过简单的PHP脚本把数据从离线的MySQL数据库抽取出来，向内部用户展示收集到的信息摘要（Summary）。这是我们第一次可以回答网站特性对用户行为的影响。早期通过以下几种渠道分析最大化增长：登出用户的默认页面的布局、邀请来源、Email联系方式导入器的设计。除了以上分析，我们开始通过历史数据开发简单的产品，包括对赞助商成员特性进行聚集的内部项目。实践证明，该项目很受品牌广告商欢迎。

我那时没有意识到，实际上，通过ETL框架、数据仓库和内部控制台，我们已经构建了一个简单的“商业智能”系统。

“猎豹”和“大象”译注1

从第一天开始对Facebook的点击流写日志起，到现在我们已经收集了超过400GB的数据。对该数据集的加载、索引和聚集操作对Oracle数据库的负载很重。虽然做了很多优化操作，但是我们还是无法在24小时内完成对一天的点击流的聚集操作。很显然，我们需要把日志文件聚集到数据库外，只在数据库中保存摘要信息供后期查询。

幸运的是，一个来自某大型网站的顶尖工程师加入了我们团队，他有过处理大规模Web点击流的经验。仅仅几周的时间，该工程师就构建了一个名为Cheetah（猎豹）的并发日志处理系统，该系统能够在两个小时内处理一天的点击流。这实在太让人振奋了。

但是，Cheetah存在一些不足：首先，在处理完点击流数据后，原始数据还是以归档方式保存，不能够被再次查询。此外，Cheetah是从一个共享的NetApp归档数据中获取点击流数据，而NetApp归档数据的读带宽受限。每个日志文件的“模式”是嵌入在处理脚本中，而不是保存为可查询格式。我们没有收集进程信息，而是通过Unix基础工具cron来调用Cheetah任务，因此无法应用复杂的加载共享逻辑。最重要的是，Cheetah不是开源的。我们团队很小，资源有限，无法分配更多的资源来开发、维护和给新用户培训使用Cheetah系统。

Apache的Hadoop项目，由Doug Cutting和Mike Cafarella于2005年末启动，是我们取代Cheetah的最佳选择。以Doug的孩子的玩具大象命名，Hadoop项目的目标是实现遵从Apache 2.0许可的G公司的分布式文件系统和MapReduce技术。雅虎在2006年1月聘用了Doug Cutting，并投入了大量的

Jeff Hammerbacher，Cloudera公司产品副经理和首席科学家，拥有哈佛大学数学专业学士学位。他在Facebook构思、创建和领导了数据组。该数据组发表了两个开源项目：Hadoop上构建的离线分析系统Hive和P2P网络上的结构化存储系统Cassandra。

工程资源来开发Hadoop。在2006年4月，该软件使用188台服务器，能够在47小时内，对1.9TB的数据进行排序。虽然Hadoop的设计在很多方面优于Cheetah，但它在那时还太慢了，不能够满足我们的需求。在2008年4月，Hadoop用910台服务器，可以在209秒内对1TB的数据进行排序。由于Hadoop性能的改进，我说服了运行组团队利用60台Web服务器和3台500GB的SATA驱动器，开始在Facebook第一次部署Hadoop集群。

在最开始，我们通过流方式在Hadoop和Cheetah中都导入一部分日志。Hadoop增强的编程能力加上其能够查询历史数据，从而推动了一些其他有趣的项目。其中一个应用是对所有Facebook用户交互的有向对进行打分来确定这些用户的亲密程度；这个分数可以被用于搜索和新闻订阅的排序。过了一段时间，我们把所有的Cheetah工作流都迁移到Hadoop上，废弃了前者。后来，事务数据库收集程序也都迁移到了Hadoop。

有了Hadoop，Facebook的基础设施可以支持对无结构化和结构化的数据的大规模分析。随着平台扩展为每天几百TB的数据规模，可以执行成千上万个任务，我们发现由于现在系统能够存储和检索的数据规模很大，我们可以构建新的应用，探索新问题的答案。

当Facebook向所有的用户开放注册，用户数在一些国家增长迅猛。但是在那时，我们无法根据国家执行点击流粒度分析。自从有了Hadoop集群，我们可以通过加载所有的历史访问日志到Hadoop，写一些简单的MapReduce任务来重新分析Facebook在一些国家，如加拿大和挪威增长迅猛的原因。

Facebook的用户每天都有几百万半公开的对话。据一次内部估算，留言板的数据量是博客的10倍！但是，这些对话的内容还是无法进行访问用来数据分析。在2007年，一个对语言学和统计学有强烈兴趣的暑期实习生Roddy Lindsay加入了数据组。通过Hadoop，Roddy能够独立构建一个强大的趋势分析系统，该系统名为Lexicon，每天晚上能够处理TB级别的留言板数据。

新工具和应用研究

在Facebook，绝大部分Hadoop集群的早期用户都是渴望追求新兴技术的工程师。为了使企业的更多人可以访问信息，我们在Hadoop上构建了一个数据仓库框架，并称为Hive。

Hive的查询语言类似于SQL，支持嵌入MapReduce逻辑、表分区、抽样和处理任意序列化数据的能力。最后一个特征至关重要，因为收集到Hadoop的数据在结构上不断变化；允许用户指定自己的序列化模式，可以使我们把为数据指定结构问题转为把数据加载到Hive。此外，我们还实现了一个简单的用户界面来构建Hive查询，名为Hipal。使用这些新的工具，市场、产品管理、销售和客户服务的非工程师都能够在几TB的数据上自己执行查询。经过几个月的内部使用后，在Apache 2.0许可下，Hive成为Hadoop的官方子系统，现在仍然在积极地开发中。

构建信息平台采用的硬件和软件将会迅速演化，并且数据科学家需要掌握的技术也将以同样的速度变化。保持致力于加速学习过程的目标对于企业组织和科学都有帮助。

在为Facebook应用构建信誉积分系统时，我们证明了把不同系统的数据存储到相同的存储库中会导致严重的问题。在2007年5月启动了Facebook平台后不久，我们的用户就被“淹没”在添加应用的请求中。我们很快意识到需要添加一个工具来识别有用的应用和用户认为是spam的应用。通过收集API服务器的数据、用户信息以及来自网站本身的行为数据，系统能够构建一个模型对应用进行打分，这使得系统可以分发我们认为对用户最有用的应用邀请。

除了Hive，我们构建了分享图表和图形的门户Argus（受IBM的Many Eyes项目启发）、工作流管理系统Databee、用Python写MapReduce脚本的框架PyHive、为终端用户提供结构化数据服务的存储系统Cassandra（现在作

为开源，在Apache孵化器中）。

随着这些新系统的稳定，我们最终构建了由单一Hadoop集群管理的多层模式的数据。企业中的所有数据，包括应用日志、事务数据库和Web爬虫，都以原始数据格式，定期收集到Hadoop分布式文件系统中。夜间执行的几万个Databee进程将把一部分数据转化为结构化格式，把它放入由Hive管理的HDFS文件目录中。在Hive中执行下一步聚集操作，用来生成Argus服务报表。此外，在HDFS内，在自己的home目录下维护“沙盒”的工程师可以运行原型任务。

目前，Hadoop包含了将近2.5PB的数据，而且以每天15TB的数量级增加。每天都有3000个以上的MapReduce任务在运行，处理55TB的数据。为了适应这些运行在集群上的任务的不同优先级，我们构建了作业调度器，实现在多个队列上的资源共享。

除了支持内部和外部的报表、a/b测试管道和很多不同的数据密集型产品和服务，Facebook的Hadoop集群可以实现一些有趣的应用研究项目。

由数据科学家Itamar Rosenn 和Cameron Marlow主持的一个纵向研究项目用于预测长期的用户参与的最重要的因素是什么。我们使用信息平台来选择一些用户的样本，删除游离点，并对参与度的不同尺度使用一些最小角度回归技术来生成大量的特性。有些特性能够通过Hadoop生成，包含计算好友网络密度的各种尺度和基于信息特性的用户范围。

另一个探索激励新用户贡献内容的动机的内部研究，在2009年CHI 会议的论文“Feed Me: Motivating Newcomer Contribution in Social Network Sites”中有描述。Facebook数据组的一个更新的研究是查看信息流是如何在Facebook的社会图中流动，该研究的标题为“Gesundheit! Modeling Contagion through Facebook News Feed”，已被2009 ICWSM会议接收。

在Facebook，每天收集证据、测试假设、构建应用和使用共享的信息平台生成新的洞察。而在Facebook之外，其他公司也同时构建了类似的系统。

数据科学家

在最近的访谈中，G公司首席经济学家Hal Varian强调了员工需要能够从之前描述的信息平台中抽取信息。正如Varian所言：“找到能够为一些变得普遍且廉价的东西提供稀缺、互补的服务。那么，是什么变得普遍且廉价？数据。是什么与数据相辅相成？分析。”

在Facebook，我们发现传统的头衔如商业分析师、统计学家、工程师和研究科学家都不能确切地定义我们团队的角色。该角色的工作是变化多样的：在任意给定的一天，团队的一个成员可以用Python实现一个多阶段的处理管道流、设计假设检验、用工具R在数据样本上执行回归测试、在Hadoop上为数据密集型产品或服务设计和实现算法，或者把我们分析的结果以清晰简洁的方式展示给企业的其他成员。为了掌握完成这多方面任务需要的技术，我们创造了“数据科学家”这种角色。

在金融服务领域已经构建了历史市场行为的大数据存储作为该领域的数据科学家，即数据分析专家（Quants），来开发新模型的实验场。在工业以外，我发现在很多科学领域，研究生扮演着数据科学家的角色。Facebook数据组团队的其中一员曾在生物信息实验室工作过，在那里他构建过数据管道流，并做类似的离线数据分析。在CERN，著名的Large Hadron Collider生成大量的数据，这些数据是由一群追求突破的研究生精心收集和钻研的。

最近新出的书如Davenport和Harris合著的《Competing on Analytics》（哈佛商学院出版社，2007），Baker的《The Numerati》（Houghton Mifflin Harcourt，2008）以及Ayres的《Super Crunchers》（Bantam，2008）都强调了在跨工业中数据科学家的重要性，他们在促进企业基于收集到的信息做出改进发挥了至关重要的作用。和研究社区在数据空间的调研一起，数据科学家在今后几年需要进一步的定义。通过更好的阐明数据科学家角色，我们可以建设培训课程、制定广告层次、组织会议、写书以及为任何被认可的行业做补充。在这个过程中，可行的数据科学家组织将会不断扩展，用来满足飞速增殖的数据平台上不断增长的专业“领航员”需求，进一步加速跨企业的学习过程。

结论

当面对在Facebook构建一个信息平台的挑战时，我发现观察别人是如何跨越时间和问题领域来解决相同的问题是很有帮助的。作为工程师，我最初的做法是通过已有可得的技术作为指导，这在现在看来显得有点目光短浅。最大的挑战是一直致力于研究构建“学习型组织”的基础平台和人员构成这个大的问题，而不是某些特定的技术系统，如数据仓库或企业搜索系统。

我确信构建信息平台采用的硬件和软件将会迅速演化，并且数据科学家需要掌握的技术也将以同样的速度变化。保持致力于加速学习过程的目标对于企业组织和科学都有帮助。未来属于数据科学家！

译注1：猎豹和大象在此采用了借代的修辞方法。猎豹（cheetah）指的是Facebook的The Cheetah日志处理系统，大象（elephant）则代指的是Hadoop项目。

本文摘选自《数据之美》（《Beautiful Data》）一书，中文版由机械工业出版社华章公司推出，特此感谢华章公司授权支持。

专访微软 MVP陈希章：MVP助我成长

■ 记者 / 杨东杰

陈希章还清楚地记得2006年第一次获得MVP荣誉的情景，他像往常一样早起，在查收邮件时，看到了微软发送过来的祝贺信。“那天正是元旦，这个消息是我收到的最好的新年礼物。”陈希章说。

尽管已是第5次连任微软MVP，陈希章回首十几年的技术发展之路，从非技术科班出身到技术行业、从向高手学习到自己成为学习的对象、从技术咨询到二次创业，MVP的相关经历仍是其中最亮丽的风景。

记者：请谈谈您的技术成长经历和心得？

陈希章：我不是计算机科班出身，在1999年出于兴趣开始自学Visual Basic，并为当时工作的公司编写过各种各样的业务程序。我还很清楚地记得用ASP做出第一个网站时的激动心情。

我的技术成长最快的就是那两三年时间，完全是基于自己的兴趣和爱好，自由地探索和学习，要知道当年的学习资源可远没有现在丰富，所以我非常认同“兴趣是最好的老师”这句话。

后来我接受了正规和权威的培训，通过了微软的专业认证（MCSD和MCDBA），也进入了专业的软件公司工作，才真正开始在.NET和SQL Server两大平台，发挥自己的技术能力。期间我还经历了一次不算成功的创业，但收获的经验却很珍贵。这次创业也使我认识到，“路是靠走出来的，要敢于梦想”。

随后的大多数时间，我作为微软特邀的顾问在一些公开场合发表演讲，我觉得这正是最适合我的工作。与此同时，我和华东地区一些合作伙伴紧密合作，给广大的企业客户提供咨询和教育培训方面的服务。

通过多年的技术咨询工作，我对企业的需求开始有了深刻了解，对中国软件行业的前景也很有信心。最近我正在准备自己的第二次创业——新在线（xinonline）服务平台，主要是依托“软件即服务”和“云计算”的理念，提供可信赖的基础架构和专业服务。

无论工作还是创业，指导我个人成长的法则一直没变，那就是“坚持做自己喜欢和擅长的事情。”

记者：最近在关注什么新的产品和技术？

陈希章：刚才我谈到正在准备第二次创业，是因为我认为我们正在经历由云计算和移动互联网技术推动的第三次互联网热潮，我希望抓住这个机遇，所以我目前关注的技术也是这两个领域，主要是微软的Windows Azure平台、Windows Phone和HTML 5。

关注HTML 5是因为它将改变Web的现状和未来。与此相关的，我也对Silverlight有足够的信心，尤其是它结合RIA Service在Business Application开发中的表现。

记者：很多人对如何在众多的微软技术中选择适合自己项目的框架和技术感到困惑，您有何建议？

多年技术咨询后，陈希章投入第二次创业

陈希章：我的体会是你需要针对不同的业务场景选择不同的框架和技术，当然没有唯一的答案，下面仅列几条方向性参考意见。

- 选择VSTO和OpenXML，而不是用VBA进行Office开发。
- 尽量利用SharePoint搭建企业内部应用，而不是每套应用都是从零开始自己做。
- 善用SQL Server提供的功能，例如在XML和BI方面的组件，避免自己去写类似的功能。
- 选择WCF，而不是XML Web Service 进行服务开发。
- 在Web Forms和MVC之间选择，主要取决于应用程序的复杂性，通常而言，MVC 适合更加大的系统，也更加易于扩展。

互联网公司的两条腿：创新 +标准化

技术团队的壮大往往会带来管理上的瓶颈，互联网公司尤其如此。北京暴风网际科技有限公司CTO杨立东认为，互联网公司在规模较小时，成功运转主要靠精神领袖的能力；当规模扩大之后，就要靠“两条腿走路”，即创新和标准化。

杨立东
北京暴风网际科技有限公司 CTO

记者：成为一名成功的互联网公司CTO，您认为需要哪些条件？

杨立东：我觉得需要具备以下三种能力：**第一**，要具备搭建团队的能力。团队的建设，要符合公司的战略构想。**第二**，要具备较强的决策力。应该对产品的技术走向有比较深入的研究，并作出决策。**第三**，要具备推动公司规范的能力。规范不是为了限制人的发挥，而是为了塑造良好的工作习惯，提升工作效率——这一点我认为最重要。互联网公司通常在不断扩大的时候才能体现自身的价值。在小公司里，团队的成功运转主要靠精神领袖的能力；但当公司规模逐渐扩大，就一定要靠“两条腿走路”。第一条腿叫做保持创新。我们不能压制员工们的创新热情。在暴风公司，就有很多创新型项目组，成员们始终在研究自己感兴趣的技术领域。第二条腿叫做运营过程管理。作为技术人员，我们希望所有的项目都能稳定运行，这就需要用一个规范的过程体系对其进行管理。

记者：现在很多年轻程序员把管理作为自己的发展目标，您怎么看？

杨立东：在开发者群体中，现在90%的年轻人都认为，应该先在公司写几年程序，然后转向管理；相反极少有人会说，他想写几年程序，然后争取做一名架构师。这说明整个行业中，程序员的心态比较浮躁，觉得年龄大了只能做管理。而我自己更倾向于纯技术的职业发展路线，即程序员→高级程序员→架构师→资深架构师。

记者：作为暴风公司的CTO，您怎样规划暴风的产品研发方向？

杨立东：目前，我们的一个工作重点是将暴风影音的版本更新到5.0，预计在今年4月底发布。新版本会在现有版本性能的基础上产生新的飞跃。我们将以KMPlayer为目标，在CPU和内存的占用、客户端启动速度等方面进行新的改进。

另一个重点是高清在线视频。我们已经翻越了高清在线播放的“三座大山”，即解决了用户带宽、服务器带宽、电脑性能的瓶颈，并成功实现了“1M带宽看720P高清电影”的在线点播产品。这项技术在国内已远远领先其他同类产品，我们会在这个产品上继续加大投入，让用户可以更简单地享受高清。此外，我们在版权方面也做出了很大的投入。目前，国家正在加大对盗版的打击力度，所以我们的片源也将全部从影片商那里购买，之前的盗版片源，都已经下线。迄今为止，清晰度为720P的影片，我们已经拥有1000多部。

记者：您目前关注哪些技术领域？

杨立东：我目前最关注云计算，并且正在进行实践性的探索。我们公司今年将会发布一款基于“云”思想的“云万能”播放产品，它会针对不同性能的机器，配置相关的播放方案。我们正在通过专门的程序收集相关的信息传到云端。我相信经过长期的积累，我们的云端会聚集海量的相关信息。这样一来，以后每当有硬件配置信息传输到云端，我们便能根据用户的具体情况，找出最适合的播放配置方案。而这期间无需用户对播放器进行配置，所有的工作由我们的产品自动来完成。

另外，对于播放器软件的未来，我不赞同浏览器会取代本地软件的说法。我认为在云时代，人们可能更不愿去记忆那些繁琐的网址，而更喜欢到Google、百度等搜索引擎上去寻找。在享受视频的过程中，用户更愿意使用播放器上的搜索功能去搜索影片。在这种情况下，具备联网功能的客户端对用户的吸引力将更大，当然其底层还是需要大量的“云”去支持它们的服务。（记者/张雪峰）

四问云 GIS

云时代的GIS软件开发商该如何生存？在Esri中国（北京）有限公司首席技术官王昊看来，存在两条生存之道：要么做深度的差异化应用，要么转型为平台的运营和服务提供者。

记者：您如何看待云计算给GIS产业带来的影响？

王昊：毋庸置疑，GIS平台和云计算的结合是未来认可的技术方向，但不会是产品的全部形态，包括客户端部署在内的多种方式在未来将继续存在。如今，国外基于大型云计算平台提供的空间信息数据和服务，在产品架构和商业模式上都已有了较好的实践；但在国内，就目前情况来看，云GIS在商业模式、用户分布、技术以及相关政策上仍面临挑战。从长远来看，GIS厂商需要新业务模式，用户需要更便捷和灵活的服务，多方需求将共同推进云计算和GIS的结合。

记者：为什么云GIS在国外的实践比国内推进得更快？

王昊：一方面，国外GIS用户中有很多都是中小企业甚至个人，他们更关注业务本身发展，对数据安全性的担忧则可以降低一个档次。Esri目前在亚马逊弹性计算云（Amazon Elastic Compute Cloud，Amazon EC2）上提供空间信息数据和服务已经有了较成熟的架构，可以在亚马逊云中部署预先配置好的ArcGIS Server以及企业级Geodatabase镜像，而用户除了遵循亚马逊的付费要求外也可以根据ArcGIS的“Term License”选择不同的租赁使用模式。

而国内当前GIS的用户主要还是政府或大型企业，这类用户各方面顾虑比较多，如果从主流客户（政府）来推进云GIS，无论是公有云还是私有云，都将是一个渐进的过程。众所周知，业内对GIS数据的安全性和保密性都有很高的政策要求。

另一方面，国内目前还缺乏像亚马逊这样较成熟的云计算基础设施服务提供商，例如国内某领先的云计算基础设施供应商，现在对虚拟镜像的支持只有屈指可数的Windows版本，机房管理模式也都跟不上。诸如此类的因素限制了云GIS模式的快速普及。

记者：云计算时代对GIS平台本身的体系架构会带来哪些冲击？

王昊：最大的改变就是GIS平台所有的技术点都将“暴露”成一个个可调用、可访问的服务，一切都是开放性的、以服务的形式展现，整个产品是一个具有松耦合、可移动、可伸缩性和自适应性的架构。必须强调，云计算绝不仅仅是局部应用模块的虚拟化，而是包括存储、数据库（空间数据库）等在内，整个基础架构都将以服务形式来提供。举例说，过去访问空间数据库都要通过特定的接口，Esri有Geodatabase的API，其他主流厂商也都有自己的格式，但在云GIS时代，这些都必须用公用的服务类型来表述。客户的各项请求包括桌面应用都将通过服务端接受远程指令的形式来得到满足。

记者：GIS软件开发商如何应对云计算浪潮？

王昊：云GIS时代ISV的生存环境大致有两个方向：第一种方向，ISV的技术和商业模式几乎不需要做任何改变，只是应用的部署方式会有所不同，过去是部署在自己的机房，现在则把应用部署到专业的（云）数据中心。第二种方向，ISV可以基于云做一种业务模型，为垂直行业用户提供基于私有云的解决方案，在这样的情况下，ISV可以转型为侧重于云平台维护和运营的服务提供者。例如在环保领域，可以做一个排污收费和地图应用相结合的产品，部署在云端，各地环保局等组织在需要的时候可以用租赁或购买的方式来获取数据和应用。（记者/付江）

王昊 Esri中国（北京）有限公司首席技术官

我的创业故事

■ 策划 / 本刊编辑部

本期封面报道，我们选择了九位开发者的创业故事。

他们经历迥异，有的插过队、下过乡；有的长年漂泊海外；有的一度是白领宅男；有的似乎总是在跳槽与创业间不断迭代；甚至还有位是老外们心目中的“活雷锋”。

他们对创业的理解与众不同，有的认为创业就是“软件工程意义的重构”，有的认为创业就是“倒着走路”，有的认为创业“不需要完美”，有的认为创业就是“勇敢的冲动”。

他们做过的产品，我们几乎可以用来勾勒整个中国软件的产业链——基础软件、企业应用、Web2.0、信息安全、移动应用、教育培训甚至传播媒体。

但无一例外的，他们都出身于程序员，他们都有着技术人员特有的理性、坚韧、透明。他们和我们一样，都曾经在电脑前敲下过“Hello World”代码；他们和我们一样，都正在同样的生存环境下，探索适合自己的发展（或创富）道路。他们是我们中间的一份子。

古人说：“九者，阳之数，道之纲纪也。”

在2011年春节到来之际，《程序员》编辑部希望这九个精心筛选的创业故事，能给广大有创业梦想、或正在创业的读者们开启一扇全新的窗口，并带来心灵的共鸣。

最后祝大家新春快乐，在新的一年里美梦成真！

做面向大众消费的产品

——记架势无线CEO叶忻

■ 记者 / 常政

说到软件创业，叶忻，绝对属于那种为数不多、同时皆备海外和中国创业经历的资深前辈了：早在1992年，他便在美国成立了自己的软件公司；随后，他还加盟过当时硅谷的新兴创业公司Marimba、TIBCO，并且也是硅谷华人创业家俱乐部最早的成员之一。回中国后，他先后在Sohu、中关村软件、掌上灵通等公司做过CTO，期间还自己创立了汉星天软件公司。而自2006年至今，应移动创富热潮，他又正式创办了架势无线，专注于移动广告平台业务。

某种意义上讲，叶忻是过去20年里中国从事IT技术的创业人员的标杆：天子骄子（清华和美国威斯康星玛凯特大学毕业生）、赶上了出国潮、创业潮，并且几乎经历了中国互联网发展的所有关键进程。而经过这么多年的创业风云和人生起伏，待尘埃落定，叶忻向广大年轻IT创业者的建议是——做面向大众消费的产品。

销售是一种生活方式

上世纪90年代初，意气风发的叶忻自美国威斯康星玛凯特大学毕业后，便在美国创立了自己的第一家软件公司Innovis，经营中美软件外包业务。尽管进展并不顺利，但让叶忻意识到了创业者需要的两个必备条件：资金和销售力。尤其是后者，对于技术背景出身的中国创业者来说，往往是一大软肋。叶忻也切身体验到了为了找订单，挨家挨户地敲美国客户大门时的那种捉襟见肘。公司最终未能经营下去，叶忻却因祸得福，凭借创业时期磨炼出来的交际能力，顺利地获得了当时硅谷最炙手可热的创新公司Marimba的Offer，专职商务销售工作。Marimba 当时基于Java 开发出了一套提供Push Technology 的产品，值得一提的是，创始人正是促使Java一举成名的原Sun公司Java项目产品经理Kim Polese；同时Marimba 还获得了凯鹏华盈的融资，成为硅谷继网景、Yahoo 之后，当时最受瞩目的技术创新公司。所以在这样的环境里从事销售，对于叶忻来说，实在是千载难逢的学习商业管理和营销的机会。

叶忻说：“在我们这一辈的海归里，我属于特别幸运的。因为在美国人的思维定势下，来自中国的IT留学生总给人一副吃苦耐劳、埋头编码的工程师形象；但我却有机会参与到硅谷的市场、销售活动去，所以这是非常难得的经历。”

有趣的是，最初去Marimba面试时，叶忻本是想应聘技术工作的，但由于其不错的仪表和谈吐，使得面试官建议他转做商务。叶忻至今还对面试官当时说的一番话印象深刻，它改变了叶忻的初衷，使他领悟到销售其实也是人生常态。面试官说：“你如果拥有销售经历，将终身受益。比如小时候在家庭里，你只有把自己销售出来，才可能在兄弟姐妹中脱颖而出，获得父母的喜欢。等进入学校，只有把自己包装出色，才可能获得老师甚至女孩子们的青睐。所以人的一生，本质都是在做销售。”

硅谷的商务经历，使得叶忻对于销售产生了全新的理解，同时也体会到了中美对于销售观念的差异：“中国人做销售，习惯宣传‘物廉价美’；而美国人更注重商品的价值营销，比如他们对知识产权保护尤为重视。”叶忻认为，如今自己和其他在国外纯做技术的海归们相比，有两大优势：“第一，我在美国创过

叶忻表示，创新的人最需要的是事业

业；第二，我很清楚一个生意的环境下，如何和不同背景的客户沟通。”

创新的人不需要稳定

硅谷给叶忻带来的重要改变，还包括对职场和事业的理解。叶忻在Marimba工作的三年里，公司一直发展得顺风顺水，并成功实现了上市，许多创业员工也因此成为百万富翁。但令许多国人困惑不已的是，叶忻却在此刻选择了离开。叶忻解释说：“其实这在硅谷是很平常的事情。除非你是创业者，否则当一家公司已经发展到巅峰后，很难有什么新的挑战可以继续激励你了，尤其公司上市后，会行走在一个固定节奏的稳定轨道上，使得你很难有进一步的提升空间；所以大家往往会选择开辟新的战场——这也是硅谷总是源源不断地出现创新企业的重要原因。”叶忻表示不太理解许多国人把追求安逸稳定的工作作为职业理想，他说：“充满创新激情的年轻人不需要稳定，他最需要的是事业。”

本着这样的理念，就不难理解叶忻后来的职业轨迹：他又在硅谷辗转了数家公司（包括以企业软件著称于世的TIBCO），随后在1999年选择了回国，并在Sohu、中关村软件、掌上灵通等公司担任首席技术官，而无一例外的，等这些公司发展成熟或无法进一步施展自己的抱负时，他便选择了离开。但这样的奔波辗转中，叶忻心灵深处的创业之火从未熄灭过，技术、产品、团队、市场、资本、经验……他一直在积累创业的底蕴。经历过汉星天软件公司的创业实践后，2006年，叶忻正式创立了“架势无线”，主打产品是移动广告应用平台。

创业要避实就虚

叶忻看好移动应用的缘由很简单，他说尽管当时还没有iPhone或Android，但大家已经开始用手机上网，这使他预感移动终端应用将是继互联网后，下一代的IT掘金地，市场潜力不可估量。同时，推出无线广告应用平台，也不是他头脑发热、一蹴而就的产物，相反经过了积极的尝试、周密的考量。

首先，在Sohu的工作经历，使得叶忻明白如果创业做手机门户，基本死路一条。叶忻说：“比如像Sohu这样的大型门户集团，拥有近300多人的媒体团队，如果他们想进军手机门户，只需做个CMS系统，便可以很轻松地生成手机门户的信息资源。所以和他们相比，白手起家的创业者没有任何竞争优势，空中网的失败已经证明了这一点。”

其次，叶忻认为做移动综合娱乐平台也很难行得通，因为前面有腾讯这只拦路虎，普通创业者无法具备它那样的用户群基础。叶忻记得在掌上灵通做CTO时，该公司曾做过这样的尝试，结果反响平平。

于是，叶忻一度看好手机搜索，甚至率队开发出了一套原型系统，但某次在威海机场的经历，使得他彻底打消了做搜索的念头。当

时，利用候机的余暇，他向一企业家朋友演示所开发的手机搜索产品。这吸引了附近的一小姑娘，她过来问了一句：“这东西真不错！能否告诉我怎么在手机上用百度？”言者无心，但对叶忻来说，犹如惊雷。叶忻感叹说：“哪怕拿一亿美元做宣传，恐怕也难以改变中国用户对百度这个搜索品牌的认知。”

叶忻认为在移动领域创业，一定要避实就虚，只有做Google、百度等商业巨头做不了或者不愿意做的服务和应用，才有机会。那做什么好呢？处在立项彷徨期的叶忻决定向业界最顶尖的精英们请教。他特地去硅谷拜访了搜索引擎先驱公司Inktomi的前研发副总裁Brian Totty。Brian Totty推荐他去观摩一下某款名叫AdMob的移动广告产品。叶忻顿时眼前一亮：“首先，AdMob的业务模式非常清晰，一看就明白；其次，容易上手，而且国内也潜在着类似的移动广告需求。”叶忻决心做中国的AdMob，并于2007年下半年正式推出架势无线的移动广告平台。当然他深知简单复制AdMob是没有意义的，必须根据中国国情做出差异化来，好比淘宝之于eBay。叶忻说：“比如手机版本的杂志应用，如果在美国只需要简单地购买流量就行了，但在中国还需要提供诸如杂志下载、订阅等附加服务。”

众所周知，中国软件业素有山寨之风，尤其是那些技术门槛低、以业务模式和服务取胜的产品，一旦有人尝到甜头，效仿者总是如过江之鲫。但叶忻并不担心竞争对手们复制他的产品，他说：“除非是Oracle的数据库引擎，或者Intel的芯片，现在软件业里，几乎没有一款新产品，另一家公司不能在六个月内复制的。对于移动广告平台，我们有中国同行们无法比拟的优势。首先，我们拥有一支具有硅谷研发水平的核心技术团队；其次，架势无线的创始成员都是业内精英，并拥有具备业界影响力的人际关系网；第三，我们已经在无线互联网领域做了多年的沉淀，并在互联网广告、搜索广告、无线广告营销等方面积累了丰富的经验。”叶忻认为这三点优势，是对手们短期内无法复制的。

在诚信匮乏的市场内生存

架势无线的移动广告平台，自推出后一直如叶忻所预计，成长迅速：目前已经与国内一万多家无线WAP网站的内容运营商进行合作；以Android平台为例，平均每日拥有100多万的PV；2010年的全年广告收入更是达到3000多万。所以在中国的移动广告市场，无论从广告量还是收入，架势无线均排名榜首。尽管形势颇佳，但叶忻坦言，前方未必都是坦途，尤其是在中国特色的国情下，如何在诚信匮乏的市场内发展下去，将是未来最大的挑战。

叶忻说：“做企业需要在一套规范的行业准则内进行。但目前的大环境下，中国企业之间总是缺乏信任，总认为对方是骗子。而我们也总遇到广告主不付钱，或者客户拖账、赖账等现象。最近我们还起诉了一家公司，它真的欺骗了我们。幸亏我们事先保留了法律记录，最终获胜。”

面对如此环境，叶忻的对策是专业化运作，以规范的服务和专业化的资金链，来保证架势无线的上下游合作伙伴们的商业利益。叶忻说：“比如一款挂了我们代码的手机应用，只要带来了流量，即使我们这边还没有收到钱，我都会付费给那个开发者。”叶忻表示能这么做的原因，是基于对中国IT行业的长远信心，他相信这个产业会慢慢规范起来。

做面向大众消费的产品

从硅谷回国至今，叶忻已经在中国IT圈摸爬滚打10多年，从企业软件到广告，从互联网到移动，几乎经历了中国软件的每一波热潮。回顾往昔，叶忻给中国广大技术创业者的建议是做面向大众消费的产品，并强调这是基于痛苦经历的经验：“首先，即使从全球角度出发，企业软件也非常难卖，因为对它的要求很高；其次，中国的企业大多数是国企，所以对于销售将面临许多挑战，比如无数中国特色的流程、沟通规则等。”而对于大众消费者市场，叶忻认为“一直非常开放”、“进入门槛低”，所以尤其适合小团队创业。叶忻透露：“最近15年回中国的海归们，凡是做企业软件或企业高科技服务的，基本死伤大半，而成功下来的公司市值比起做大众消费者的市值，仅有人家的1/10。”

坚持与展望

——上海科泰华捷科技有限公司董事长陈榕专访

■ 记者 / 张蜀

记者手记：1977年高考的大学生意味着什么？让我想起路遥的一篇小说《人生》里的男主人公，当年的天之骄子啊。我较陈榕先生晚几年入清华，当年可能在学校见到过，但没有机会相识。77届学生对我来说挺神秘的，我从他们那里学到了很多东西，有时甚至超过了大学老师们所传授的知识，比如大学时没能理解微积分，相反给我印象最深的却是一位77届学生激情四溢讲解的斯特拉文斯基《春之祭》。而我在几十年之后，采访一位77届学长，也算是一个报答吧。

记者：您好，我记得《程序员》杂志以前采访过您。

陈榕：对。《程序员》杂志社8~9年前邀请过C++的发明者Bjarne Stroustrup来清华和北大讲演，那次我自告奋勇去当翻译，顺便解释了一些C++背景方面的故事，大家都比较有兴趣。我1986年在美国读书的时候就开始学习C++了，那时我所在的伊利诺大学正在设计世界上第一个用C++编写的操作系统Choices。《设计模式》的四位作者之一Ralph Johnson，他那时很年轻，一副娃娃脸，刚博士毕业，来伊利诺教书。我们跟他学习了Smalltalk和面向对象的基础知识，当年觉得面向对象可能就是Silver Bullet，我曾经花了很大工夫去研究面向对象和操作系统。

记者：我也对语言很感兴趣。我认为metaphysics永远不具有physics的丰富性，所以我更喜欢非理性的语言，比如像Perl和C。metaphysics只是一种封装，我想这是C++的问题。不过这次采访对象不是我，还是谈您吧，后来您怎么想到创业？

陈榕：其实我的创业非常不典型，做的事情不典型，年龄也不典型，只能说是“非典型创业”，对读者不一定有帮助，就当故事听吧。现在创业一般都是二、三十岁，我创业的时候已经四十二、三岁了。我1982年大学毕业后，考上中科院计算所倪光南和竺迺刚老师的出国研究生。倪老师所在的第六研究室当时正在做汉卡。设计汉卡需要硬件知识，也需要对自然语言的语法分析，所以考试题目包括大规模集成电路和编译原理。编译原理是软件专业比较难的课，当时学硬件的人一般不学编译原理。我上大学时比较用功，清华软件、硬件专业的课程都学了，所以就考上了。

我于1984年1月去了美国，1985年1月转学去了美国伊利诺大学（UIUC）。伊利诺大学在巨型计算机方面的研究是强项，我在那里学习了并行计算和计算机体系结构，还读到Amdahl's Law。受其影响，我觉得并行计算可能应用面比较窄，出路不大，便在1986年去钻研系统软件。当时我对两个领域尤其感兴趣，一个是操作系统，一个是编程模型。这也是我1992年能进入美国微软研究院操作系统组的原因。1995年我在微软浏览器IE3组的时候也参与了一些ActiveX的设计工作，后来做过DCOM。你知道OLE Automation吗？当年就是以Doug Franklin和我两人为主写的。我所有的职业生涯都和操作系统及编程模型相关，25年来始终不渝。

至于创业，确实是很偶然的机会。1998年我们Base COM组开始设计COM+，1999年跟着Win2000一起Beta，不久就被告知COM+没戏了，只做到1.0版，之后全体转移去做.NET。我们.NET组当时的口号是“Software as a Service”。但是为了实现.NET的远大目标，微软内部产生了严重的“路线斗争”。一派认为要集中全部力量，实现一个类似Java的中间代码语言，进而实现SaaS；另一派认为要兵分两路，一路做中间代码，另一路从COM层重新做起。最终第一种方案获

陈榕表示，科泰华捷的产品目标是做通用性的中间件

胜，微软决定采用C#来实现下一代OS，同时全盘放弃COM+。微软直到2004年才意识到1999年的决策失误，并重新启动了Silverlight计划，然而已经贻误了5年战机，并损失了全部COM团队。

由于当时我们用C语言实现SaaS的方案被否定了，自己觉得非常不爽，于是我就回国度假散心。那是1999年10月，美国到处都在.com（此com非彼COM）上烧钱创业，中国也在发烧，生怕钱烧不完。我在北京碰到几个朋友，他们帮忙介绍的投资人，看上了我在微软做过操作系统的背景，这样我就开始了自己的创业，可谓“逼上梁山”。

记者：怎么想到做操作系统？

陈榕：2000年时国内一般就是做做网站，并且也诞生了不少成功的互联网公司，而做基础软件，比如操作系统、中间件、数据库的人就非常少。可我不会做别的，只会做操作系统，所以没什么好想的。回头看，因为操作系统并不直接面对大众市场，所以类似项目非常不适宜创业。因为中国没有大型软件公司来收购创新型软件公司的环境，导致操作系统项目尤其不适宜在中国创业，所以我说我是非典型创业。

我要是十年前知道这些，我都要劝自己别冒傻气。但反过来说，我自1976年2月起下乡插队两年，又幸运考上大学，属于机遇不错的人。我们那代人与现在一般创业的年轻人可能不同：一是不怕吃苦、意志比较坚强，二是总想做一点此生无憾的事情。所以只要开始，就绝不轻言放弃。

记者：不过IT界很多人做的事情都是非典型的，我记得乔布斯也做过NeXT。

陈榕：说到乔布斯，其实我的经历和他的NeXT还有一些交集。当乔布斯投资NeXT Computer时，他已经被苹果开除了。再往前追溯，是20世纪80年代末，曾经有一个比较出名的微内核操作系统Mach，由卡内基·梅隆大学的一支团队发明，其主要发明人是Rick Rashid，目前是微软负责科研的高级副总裁，也是我在微软研究院时的顶头上司。Mach 1.0、2.0的主要代码实现者是Rick的一名非常出色的博士生。后来这个学生加盟NeXT，以Mach为基础，设计了NeXT OS。再后来乔布斯在苹果公司复辟，NeXT OS演变成了今天的Mac OS X。1991年下半年到1994年，大部分Mach发明团队都随Rick Rashid加盟微软研究院的操作系统组，那正是我当时（1992~1993年）工作的部门。当时除了我一个来自伊利诺大学，其他同事都来自卡内基·梅隆。

说到做操作系统，我们首先要谈什么是操作系统？比如Unix，能把进程、锁、内存都做好了，你就可以把它叫做操作系统内核。实际上操作系统就是一个运行平台。设计一个新操作系统的最主要目的，是看你要支持什么样的应用。这个着眼点，需要你对语言，即编程模型非常熟悉。所以在设计操作系统的同时，经常要设计语言。比如：设计Unix的同时设计C语言；设计.NET的同时设计C#。使用操作系统编程的人，要熟悉什么是进程、什么是线程、什么是锁这些东西。但是设计操作系统的功夫，要远远发力在操作系统之外。设计语言，目的是便于编写新型应用，而新型操作系统就是要将其跑起来。做操作系统，做的是什么？某种意义上说，操作系统就是运行平台，最早是Unix，然后是Windows，后来是.NET、WebOS，现在又将是移动互联网。至于改进一下Linux内核，并不是没有改进余地，但那已经不是问题的关键。相对IBM主机OS，Windows曾经短小精悍，是个“白领”就能上手，但Windows已经非常不适合今天的移动互联网了，病毒泛滥，安装、维护复杂，一般老百姓怎么能搞得懂？这样的情形下，我们再设计操作系统的动力将是什么呢？是新的看法、新的思想。

记者：你如何看待构件技术的应用前景？

陈榕：我最近写了一篇文章《另类云计算，另类物联网》，其中我强调了两个方面：对于我们现在这个工业化社会，一般事物可以有两种状态，一个是动态，另一个是静态，后者也可以说是零件化的。考察一下现代汽车工业，为什么它能够这样飞速地发展？零件化生产是一个重要因素。一个著名品牌的汽车，很多零件是可以互换的。在计算机软件领域里面，可不可以同样生成这样的零件？我觉得零件可以看成一个个集成电路，如果有了一堆这样的集成电路，怎样把它们动态组装起来，就是电路板的功能了，我们可以用Perl、Python、JavaScript等做这样的事情。所有这一切，零件是必要条件，它是用C/C++或者Java写不是很重

要，关键是要像一个集成电路，具有通用性、一般性。如果我们能做成功这件事，就相当于完成了构件化。

从面向对象走向面向构件：微软做的COM，IBM的SOM，或者OMG的CORBA，都是基于这样的思想。这种思想，距离它们的诞生，其实已经存在二十多年了。只是它们当年的PC战场已经灰飞烟灭，以至于人们似乎有一种错觉：它们已经被人忘记了。如同很久以前就有人在曼哈顿建了摩天大厦，但是现在居民们都移居到上海了，于是人们忘记了在曼哈顿建摩天大厦的技术其实并没有过时。而从计算机技术领域来说，现在进入了移动互联网的时代，相当于我们进入了一个新的战场。或许今天正是我们重新发现构件技术价值的机会。

记者：你们公司的产品一直定位在做操作系统上，但我看你们公司网页宣传比较多的是中间件？

陈榕：其实这两个东西，我们都没有刻意去宣传。什么是操作系统？说起来这是一个哲学问题，如我刚才所说，操作系统就是一个Runtime，三十年前Kernel是操作系统，二十年前Windows是操作系统——Windows就架在Kernel之上了，十年前.NET就是操作系统，现在是XML/HTML+JS+Plug-in。所以概念都是与时俱进的。说到中间件，也要看中间件的含义是什么。

记者：据我所知，在Java编程领域中有很多情况下用到中间件。

陈榕：你说的是Framework，它不是构件。Framework可以看成是钢梁，构件则相当于那些预制件。构件镶嵌在Framework中，你有Framework，但没有构件，或者有构件，而没有Framework，都不行。现在大家讲的中间件多是Framework，为什么不说它们是构件？因为它们不是为第三方做的。构件要做到可以替换成第三方的产品，这才能实现软件的工业化生产。Windows可以看作一个中间件，它有两个特点：首先是通用性，如它的消息驱动机制；其次是它的效率，它能嵌入CPU汇编指令，在某个时间片跑出CPU的效率。Java具有通用性，但它跑不出CPU的效率。我们再思考一下，中间件能不能做得更具一般性？我们可以用构件拼装，我们可以同步调用你，也可以异步调用，消息也是同步或者异步的。也就是说有四个接口：同步出、同步入，异步出、异步入。所以我们的产品目标是一个具有通用性的中间件。你也可以管它叫操作系统级中间件。另外我们正努力的目标是做一个具有云计算功能的中间件，它既能跑本地程序也能跑网络程序。

记者：我总结一下，你们产品的特点，一个是通用性，一个是充分利用CPU效率，还有正在努力做的是网络应用。

陈榕：是。读者可以从www.elastos.com下载我们设计的Elastos SDK。Elastos的架构及API都是我们自己设计并实现的，包括内核、图形系统、文件系统、浏览器等，花了十年时间，一千人年工作量，走了N多弯路，已经商用。一个字：难！

记者：陈先生，我想问您最后一个问题，您认为现在哪个方向创业比较有希望？并且为我们展望一下未来。

陈榕：我不鼓励大家创业时来做中间件或系统软件行业。这对个人创业者可能不是很适合。相反移动互联网绝对是一个大的浪潮，我相信这个浪潮往短里说至少也得有二十年。IBM主机时代的软件相比PC时代的软件大小总和，可以说只是“噪音”。PC时代的软件大小总和，相比移动互联网时代的消费类电子软件，我相信，也可以说只是“噪音”。也就是说，IT产业的第三次浪潮才刚开始，移动互联网时代要把几乎全部PC时代的代码都要重写几个来回，再外加全新的代码，所以创业的机会和就业的机会都会非常好。

软件工业化生产的时代正在变为现实。如果我们预想：未来的构件服务=构件+自描述信息、云编程=服务构件+拼装脚本。所以云上下雨，自然不能像原来安装Office软件那样，一坨就是二、三百MB。一条彩信300KB，那么最大的“雹子”就是300KB。再加上，雨必须从不同的云上下来。雨点可能就是一个一个功能（服务）构件、编译好的二进制代码，可能是CPU指令，也可能是Byte-Code。而HTML/XML/ JS等脚本语言要在客户端动态地将服务构件拼装成应用。所以人们将进入这样的时代——为一篇写得好的小说付钱，而不用为了印刷技术和纸张付钱。

在坚持中学会妥协

——记友友系统CEO姚宏宇

■ 记者 / 谭茂

记者手记：看姚宏宇泡茶的确是一种享受。他熟练迅速，简单而自然地完成一大套令人眼花缭乱的工序：烫壶，烫杯，闻香，置茶，分茶，水花叮咚，不走涓滴，然后亲切地对你说一声“请”，一瞬间，记者心头暖暖的，就像面对着一位邻家兄长。而姚宏宇的创业过程也一如一碗清茶，苦后回甘，亦苦亦甜。

姚宏宇表示，现在他还在坚持写代码

2000年初，在互联网的第一波浪潮时，拥有若干学位的姚宏宇进入了如日中天的雅虎，在雅虎研究院做高级研究员的经历也让他受益匪浅。他在这里既可天马行空地思考，又有机会帮助解决雅虎各部门提交上来的难题。这些难题都是在大规模分布式系统上出现的实际问题，涉及到了后台、搜索、BI等方面。这为他以后酝酿回国做一家与云计算技术相关的公司奠定了坚实的技术基础。

2004年，姚宏宇负责雅虎财经，这时候的他，除了掌管整个数据中心的运维外，还负责业务的开拓与发展。

雅虎的股票交易数据是通过雅虎全球的3大数据中心进行处理的，再实时同步发往几十个国家和地区，给那里的用户提供服务。在姚宏宇看来，这就是一大片“云”，也就是“云计算”的雏形。

2004年，时任雅虎财经技术主管的姚宏宇给杨致远建议，雅虎应该将这部分业务单独独立出来，但由于此项业务与雅虎的商业模式不符合，因此姚宏宇的这项提议也不了了之，但正是从那时候起，姚宏宇的创业心思开始萌动。

2007年，云计算的浪潮开始风起云涌，姚宏宇坚信，在未来10~20年，基于大规模分布式计算的云计算将改变整个IT产业，这对中国而言，是一个巨大的机会。这时候的他，从硅谷带着6位顶尖高手回中关村创立了专注于云计算研究的技术公司——友友系统。

选择

茶道之精髓在于选茶、选水、择器、程序——择善而从，经营之道亦复如是。

2008年初，创业时间不到半年的姚宏宇遇到了第一个选择题——技术发展方向的选择。

这正如所有人创业之初的感受——机会很多，但不知什么才是适合自己的方向，做项目还是做产品？

“在技术领域里，做项目和做产品最根本的区别是代码会重复使用多少次。重复使用的越多，你的生产效率就越高，好的产品设计出来后，可以卖100次甚至1000次，而做项目就很苦了，因为每次都是从零开始。”姚宏宇说。

经过了半年的摸索，姚宏宇同他的创业团队终于下定了决心，并明确了友友系统的发展方向，那就是做基于云计算的基础技术平台——在保护企业已有投资的前提下，帮助企业将原有的应用平滑地迁移到云计算平台上。

“其实这个方向我们一直有，但是并没有把这个作为唯一的方向，而那时候做出这个决定还是很痛苦的。”在姚宏宇看来，当时的痛苦选择有两层意思，一是放弃了有可能快速赚钱的项目，二是这个选择也意味着破釜沉舟，对友友系统而言，这是一条没有回头路的选择。

幸运的是，时间证明友友系统选择的这个

方向是正确的，分布式计算、Hadoop、海量数据处理……这些技术现已成为云计算技术领域最热门的词汇。

姚宏宇认为，云计算改变了未来，乱世出英雄，这样大家的起跑线才不会差得太远，如果现在去选择做数据库、传统的操作系统等，技术上早就落后20年了，如果没有比IBM、Oracle、微软这些公司更强大的研发能力，这个鸿沟是无法跨越的。

“我原来是做互联网的，回国后很多人很奇怪：你为什么不做互联网？”对此，姚宏宇的解释是，“互联网我玩不过年轻人，但我准备去做的东西年轻人跟我竞争会有难度，因为它有很高的门槛，所以你要找自己擅长的东西，别拼命硬碰自己的短处去跟人家的长处竞争。”

“最后能成功的公司只有两种类型，一种是自己有很强的技术壁垒，别人很难超越，这也是友友系统努力的方向，而像一些互联网公司，他们的发展原则是要能很快地把用户群扩大到一定规模，但这个并不是我们的强项。”

此外，对于初创型公司，姚宏宇认为风投的选择也是一个很大的学问：“对于风投，必须要有所选择。”姚宏宇表示，仅仅是来注资的风投是不合格的，没有客户基础的技术发展注定会失败，因此在VC的选择上，姚宏宇对风投的要求增加了一个条件，必须要有“资源”：“要想推动真正的云计算发展，到现在我还是认为国家力量将起主导作用，如果没有国家的力量在后面推动的话，新的技术是很难发展的。”

坚持

回到国内的姚宏宇，最开始创业的时候由于资金有限，只好在中关村附近租了一间民宅，连住带办公。

回首这段创业经历时，姚宏宇感慨万千：“条件虽然苦了点，但好在坚持下来了。”

“我带领的这帮创业团队，都是技术人员出身，他们都有好的想法，但要把一个好的想法变成现实，是很难的事，因为从算法上来看这些都是公开的，很多技术20年前就有了，而如果要把算法变成真正的软件，才是挑战。”

“在工作内容上，我给大家强调的是细节。”他表示：“大家都知道，细节是魔鬼，很少有人会从细节上去研究，因为细节很难说每一个都一样，细节是我们自身的问题，它是决定成功还是失败的关键。云计算技术也是一样，算法也如此，等应用的服务器规模大了一个数量级，或者再增加到一个数量级后应该怎样去做系统，它们面临的问题将完全不一样。”

“因此很多时候，我需要亲力亲为。”姚宏宇告诉笔者，他现在还在坚持写代码，最多的时候一天要写4、5个小时，对于写代码的原因，他认为主要出于技术情节和公司愿景。

作为一个技术公司的领军人物，需要时刻保持个人技术魅力的重要性不言而喻，而更重要的原因在于他需要将自己这么多年的研发经验体现到产品中去。姚宏宇表示：“友友系统现在的产品，核心引擎的开发都是基于我以前公司的经验和技术，现在的友友系统不是说没人能写，但能够写出这个引擎的确是需要经验的，而不管是经验还是技术，我对自己充满了信心。”

“我们从成立的那一刻起，就选择了和巨人们对决。在未来10~20年，基于大规模分布式计算的云计算将改变整个IT产业，这对中国而言，是一个巨大的机会，也是友友系统的机会。”

事实证明，这份坚持也给姚宏宇带来了丰厚的回报。

2008年，当默默无闻的友友系统将产品卖进了华尔街的股票交易系统时，开始有越来越多的人注意到了这家公司。而目前国内的一些央企已经开始使用云计算架构来改进自己的系统。“中国的做事方式是，先试点一个，再到一个数据中心，再到整个架构，目前友友系统的系统也开始渗入到电力、电信行业中去。”姚宏宇说。

妥协

过去的几年，尽管友友系统一直在高速发展，但也并非一帆风顺。

“有坚持，也有妥协。”姚宏宇表示：“按照我们原来的计划是希望慢慢培育这个市场，慢慢地做技术。”他把友友系统比喻为专门解决疑难杂症的老中医，自己开一个小诊所，用时间去打造属于自己的品牌。但是他很快发现，越来越多的人开始盯住这个行业，这当中既有华为、中兴这样不缺资金也不乏技术

的巨无霸，也不乏一个又一个的创业型公司。所以对友友系统而言小步慢跑已不可能，必须大踏步前进。姚宏宇指出：“诊所已经不够用了，我们需要医院。”

但是做医院是需要大量资金的，这时候就必须要引入风投。

“曾经我对风投持保守态度，因为引入风投可能会给公司运营带来一定的影响，所以才会从白手起家开始做。”对于风投的态度，姚宏宇笑了笑：“但是公司的发展需要新血液，需要新的动力，如果没有妥协，友友系统也不可能坚持到现在。”

“比如上市计划，从我个人而言，不太关注这个，但是从投资人的角度、对员工负责的角度来看，我们必须去做，这可能要花上三五年时间，这样我们也能给员工一个享受胜利果实的机会，而对投资人来说也有个退出的机制。”姚宏宇说。

对于风投带来的好处，姚宏宇的评价是风投带来了信心：“企业在一开始时，干劲很足，但两年是一个坎，一帮老员工跟着你干了两年了，看起来也没什么太大起色大家就会有疲惫，这时候如果能找到一个给力的风投进来，会给整个团队带来新的希望、新的气象、新的力量。”

姚宏宇说：“创业的经历让我明白了什么是要坚持的，同时坚持并不等于顽固，技术公司还要学会妥协，需要时刻纠正自己的错误。互联网公司有更高形态的商业模式，这也是当初雅虎不愿意去做新业务的原因，市场的未来谁也估计不到，没人能想到云计算会发展这么快，但有见识的人会很快纠正自己的错误。”

对初创人员的建议：千万不要迷信技术

如何塑造企业的核心竞争力，时刻保持技术的先进性，这是摆在所有技术型公司面前的问题。

在姚宏宇看来，从互联网公司诞生出来的云计算技术，改变了整个IT技术架构，但目前掌握这个技术的还只是几家最大的互联网公司，比如Google、Yahoo、Amazon和Facebook。因此，从这个意义上来讲，中国的软件企业同国外的软件企业是站在了同一条起跑线上。

姚宏宇：“从实质上而言，云计算技术对产业是具有破坏性的，IBM、微软、Oracle这些公司目前是陷入了两难的境地，全世界90%的软件市场已经被他们占领了，理论上不应该再有任何创新，但是互联网打破了原来的格局，大家开始换游戏玩，要想得跟上这个节奏，就必须创新。”

他表示，互联网的特点在于用户的爆发性增长很快，任何的商业模式、任何技术都不是一成不变的，一旦技术跟不上就需要去开发，开发一段时候后又去做商业模式，寻找到新的出路后又去做技术，如此反复。

“对技术人员而言，所谓的开放也是有其两面性，除非是真正的开源公司，没有真正的技术公司会把自己最重要的部分拿出来。”姚宏宇表示，“没有前瞻性的技术公司，是很难生存的，将来友友系统也会开放技术，也会走这条路。”

姚宏宇指出，云的技术门槛很高，尤其是在中国，大家都在赶时间，对所有希望从事云计算产业的初创者而言，当前最迫切的问题是先把市场份额占上，技术以后再来逐渐完善。

采访后记：选择、坚持与妥协

专注于基础软件的发展，对所有的创业公司而言，这无疑是一场赌博，毕竟过去鲜有成功的案例，但姚宏宇坚持下来了，并且取得了初步成功。

但并非所有的初创型公司都有姚宏宇这样的好运。

在笔者看来，在技术创业之路上，有的人创业之路越走越宽；有的项目换了主人，但创意还在继续；有的为迎合市场，创意做了微调。

但他们也有共同点，正如姚宏宇所言：“尽管中国目前的生存环境还是很恶劣，但不可否认的是，世界的重心正在改变，技术正在从西方向东方转移，尽管这个过程会很缓慢。如果把云计算的大环境比作天时，地利就是中国，那么人和就在于信心。我们一直在坚持，并在坚持中学会了妥协。”

创业是一种重构

——记译言、东西网创始人赵嘉敏

■ 记者 / 常政

不满足于按部就班的职场生涯和可预见的人生轨迹，34岁的Oracle工程师、美国南加州大学运筹学博士赵嘉敏想创业了却又不知道该做什么……千万别以为我在跟你讲述一则千篇一律的“宅男程序员跨越职业迷途”的故事。仅凭赵嘉敏对创业的理解便凸显出某种异化：他觉得创业并非是对过去的颠覆，而是软件工程意义上的“重构”；他行事秉承运筹学的法则——不求最优解，只做可行解。

如此精密推演的结果，导致他去做了一件看似比软件研发更加枯燥乏味的事情——IT文章翻译。然而令所有人始料未及的是，就在他真的一步一步以如此看似迂憨的方式前行后，短短三年左右时间里，成千上万的国人们因为他的步履而获得了一种全新的精神视野，甚至中国的互联网产业也由此引入了一种全新的运作模式。

不求最优解，只做可行解

赵嘉敏首次产生创业的萌动大概是2005年左右。他首先想做和自己背景相关的企业软件，但回国考察一番后发现时机不成熟。而此时国内某著名OA厂商的创始人向他抛来了橄榄枝，邀请他担任架构师（或者CTO），这着实吓了他一跳。“因为国外，一个架构师一般至少要有10年左右的时间：你需要一点点做起来，并对各种产品了解地很广、很深，才能符合架构师的要求——但国内整个的技术职业路线被大大缩短了。我当时才仅仅工作两年。”

赵嘉敏于是婉拒了这次盛邀，在他看来，如同人生捷径般的架构师职位看似“最优”，实际并不靠谱。然而当下的可行道路，即在Oracle公司里按部就班地“一年一升”、10年后做一名架构师并不是他喜欢的。“30岁以前，我一直向往那种平稳的生活。但真的过了这种生活之后，慢慢发现自己并不甘于这种简单过

赵嘉敏行事风格，秉承运筹学法则：不求最优解，只做可行解

日子的状态，一番想做点事业的愿望，让自己心里长草。”

于是赵嘉敏决定另寻可行解，但最终的选择却更加平淡无奇——英文翻译。而其缘由也很寻常，赵嘉敏回忆说：“当时（2006年7月）和两位同样在硅谷、有创业梦想的朋友张雷、赵恺聊天，商量可以做点什么事。其中一个提议说至少可以翻译一些东西来给中文读者看。我们都觉得可行，便建了一个叫‘言多必得’的翻译博客（即译言网的前身）。”

某种意义上说，这个BLOG成了赵嘉敏们宣泄创业欲望的出口。赵嘉敏坦言：“当时大家并没有真的把它当作一创业项目，更没预料到后来会产生那么大的社会影响力，只是大家都用一副创业的心态，认真地经营着这个博客。”

“言多必得”最初的翻译主题集中在IT创业。以三名硅谷工程师的学历、资历，操刀这类翻译显然是游刃有余。没多久，颇具品质的翻译水准，使得这个博客陆续汇集了不少关注者，随着他们纷纷加入翻译团队，这个博客渐渐演变成一个协作翻译社区；而随着译者们志趣的多元化，“言多必得”的译文主题开始广

泛渗透到科技、政治、经济、文化等领域中。而在国内中文网站开放性有限的大环境下，“言多必得”所展示的生动活泼的“外文世界”和开放分享的价值观，尤其是给中国的年轻一代，带来一种别样的知识和精神视野。于是这个博客以网络社区特有的口碑传播方式，开始吸引更加多的网友。

为了满足网友们日益增长的访问需求，2006年底，博客正式升级成为网站——译言网，并选择与国内的陈昊芝等合伙人进行合作，将主机正式落户在北京。至2007年2月，译言网每日的访问量已经稳定在1000 PV，网友们对内容的贡献已经远远超过了三位创业人员，这标志着译言网已渐成气候。

2007年4月，赵嘉敏辞去了在Oracle的工作，并在1个月后回到国内。回国前，他向另外两位创始人表示，应该在两年时间内把译言做起来。但整个2007年，赵嘉敏投入译言网的时间并不多，他解释道：“我的技术背景让大家觉得不适合做管理；再加上我离开中国7年了，需要一点适应的时间，所以便先去朋友开的创业公司帮忙。”

技术到管理的重构

但最终赵嘉敏还是走上了译言网的管理岗位。这缘于译言网尽管社会影响力日趋渐长，但始终不能有实质的突破：另两位创始人远在海外，国内当时负责运营的陈昊芝正在经营一个SNS的创业项目，因此译言的发展在2007年进入了一个平台期。赵嘉敏回忆说：“等到了2007年10月时，所有的股东，包括其他两位创始人，都对译言的前景不乐观。记得当时张雷给我电话，叹息说‘译言也就这样了’。”张雷的话使赵嘉敏决心回到译言，他说：“译言三个创始人，在国内的只有我一个。而我没怎么努力，就把译言放弃了，实在有点说不过去。”

于是，赵嘉敏向所有股东毛遂自荐，决心把译言真正做起来。2007年底，赵嘉敏开始接手译言的经营和管理，2008年4月正式担任总经理。

赵嘉敏并不认为自己是管理的“门外汉”：“管理译言，其实跟我的运筹学专业联系很密切。运筹学既可以用于工业流程的优化，也可以用于商业运营的管理。总之，它就是一门关于优化和管理的学问。所以技术和管理并不截然分割。”显然，这种职业角色的变化，对于赵嘉敏来说，如同软件工程意义上的“重构”：抽象级的内核功能没变，变化的只是应用层面的实现方式。

赵嘉敏表示，在他主持译言网工作期间，主要做了两方面的工作：首先，搭建出一支译言的骨干团队；其次，推进译言的商业化运作。对于前者，赵嘉敏大概花了半年时间，他说：“我刚接手译言时，它可以说是一穷二白，没有正式的办公地点，也没有团队。我记得译言网总编张文武是我招进来的第一名正式员工。随后我们向国家申请到10万元创业基金，并将办公地搬到了上地创业园。”等到2008年6月左右，随着师北宸（推广）、张国强（商务）、范欣（产品）等人的陆续加盟，译言网核心团队终于成型，正是他们决定了我们今天所看到的译言网。

众包模式的奇迹

2008年下半年，赵嘉敏决心腾出手来推进译言的商业化，但遇到了始料未及的阻力。由于译言的内容主要源于社区的无偿协作，所以一直被大众赋予了“公益”的形象，尤其是汶川地震后，译言组织网友们，以分包的模式翻译了《地震搜救手册》，更是令“译言Wiki”的公益形象广为人知。如今一旦商业化运作，难免会有人担心：这会不会使其公益形象受损，进而影响网友们协作翻译的热情？

赵嘉敏说：“另外两个创始人都觉得不着急推进商业化，但我就在国内，更能深切感受到生存才是迫在眉睫的。”赵嘉敏认为如果走纯公益路线，由于中国政府对民间公益组织的准入严格控制，而译言网本身又以公司身份注册，很难获得相关的法律保障，所以商业化才是唯一可行的出路。于是赵嘉敏率队进行了一系列商业化的尝试，包括和英语培训机构合作等，但由于没有考虑清楚译言社区的核心价值，都不太成功。

2009年对于译言网是个转折点，2月译言股东会议正式确认了商业化的发展路线。与此同时，赵嘉敏提出了“有偿众包”的盈利模式。他说：“众包，即用社区的力量去选题、翻译，并把成果提供给相关客户，然后再把报酬返回给社区分享。这就是译言后来一步一步明确出来的承

包模式。”实践证明这种模式十分契合译言的社区特质，很快使得译言网的现金流持续增长：译言网在2009年初的时候还是零收入，但到七、八月份时，每月净支出已缩小到两万元左右，而且通过基金申请，积累了20万的现金，这预示着译言在经营上正式步入正轨。

赵嘉敏缘何能产生“众包”的灵感？据他回忆，最初接触这个理念，正是源自他亲自翻译的一篇文章：Wiki创始人Jimmy Wales 所撰写的《Web 2.0 五准则》。

在赵嘉敏亲自牵头的无数众包项目中，情有独钟的当属《失控》（凯文·凯利著）一书的翻译了。2008年，赵嘉敏初次读到这部科技著作时，便爱不释手：“无论本科时学的控制学、还是博士学的运筹学，甚至创业的经历，无不在书中一一得到验证，有种精神升华的感觉。”

而《失控》的翻译，更是体现了典型的互联网众包风格。《失控》起初只有一名译者，但半年过去仅完成四分之一。于是赵嘉敏通过网上招聘来实施“众包”，结果加上校对，陆续来了10多个译者，而且很多是非专业人士。赵嘉敏说：“这些译者动机不一，有的是想做分享，有的是为提高英语水平，还有的仅为了积累翻译的资历等。总体来说，大家都把商业利益放在次要的地位，甚至我们跟这些译者连正式的翻译合同也没有。”然而结果是，这支看似松散的翻译队伍，仅用一个半月的时间完成了著作的翻译，又仅用了一个半月完成了互校。

这看似天方夜谭，一群背景迥异的人，松散地凑一起，怎么就凝聚出如此惊人的工作效率？真相当然没有表象那么简单。据赵嘉敏的分析，众包模式的成功，需要具备以下三个条件。

第一，要拥有一个沟通渠道，确保众包成员们随时无缝沟通。由于互联网的存在，这个最基本的条件目前已经具备。

第二，要拥有一种方法，使得在社区大规模协作的前提下能有效地组织生产力。赵嘉敏的解决方案是“层级架构”。这是一种自下而上的分布式控制架构：由于传统自下而上的社会协作模式（类似民主），往往带来效率的低下；所以需要在这类传统的民主协作模式上加入一个“控制层”，而这个控制层的控制范围被严格限制，不能控制底层的具体行为。以《失控》翻译团队为例，这是一个二级的层级架构，赵嘉敏的作用如同“控制层”，但他只能决定译者的人选和目标，不能干涉译者对于目标的具体实现方式。

第三，要对“众包”的基本价值有统一认识。赵嘉敏表示，目前大家对众包主要有两种误区：

首先，认为“众包就是利用廉价劳动力”。而赵嘉敏认为：“以产品设计的众包项目为例，专业团队的报价之所以高，是基于以往经验、口碑和信誉的积累，可以说，正是这些积累产生了成本；而网上的接包者，不乏水平很高者，但往往没有品牌专业团队的积累成本，所以价格自然低下来了。”

其次，认为“接包者非专业人士众多，会造成成果的不确定性和整体质量的下降”。赵嘉敏认为对于这一观念需要辩证对待：一方面，众包团队要主动去降低这种不确定性，“层级架构”就是一种很好的解决思路；另一方面，我们可以去利用这种不确定性进行创新。赵嘉敏说：“以《失控》的翻译为例，的确有很多不确定性，但正是因为这样，才会出现译者们自发地做出400多个注解的事情，这在传统的翻译流程里是不可能产生的。”

科技巨著《失控》的成功翻译，完美地凸显出社区众包模式的力量

出走译言网

2009年下半年，使译言网进入良性运转的赵嘉敏踌躇满志，准备在社区建设上大干一场，孰料天有不测之风云，很快爆发了令业界侧目的“译言网内讧”事件。这个事件直接导致了赵嘉敏的离职。回忆这段经历，赵嘉敏至今黯然神伤：“我和陈昊芝的分歧，并不像外界报道的那样，是‘我要做公益，他要做商业’的路线分歧。”

赵嘉敏如此描述他离开译言的始末：“记得2009年7月，陈昊芝突然要求我做付费阅读，否则就把运营权交给他。我当时还为此专门给全体股东写了一份报告，从版权、社区风险、模式等方面分析不能立刻上马付费阅读，但陈昊芝固执己见。于是我说可以交出运营权，但希望去负责外媒、版权项目。孰料陈昊芝拿走运营权之后，全盘否定当初的承诺，无论是版权，还是外媒均不让我碰了，甚至公司的管理层会议也不通知我参加。这样，我只能去做译言的《卫报》项目，当时也没觉得什么，毕竟这是译言的品牌项目。然而到11月份，陈昊芝

突然通知我说，如果《卫报》项目不能提三倍的报价，也要停掉，而且就算提价三倍，也要换人运营。”赵嘉敏感觉自己留在译言已经无法做事，便选择了离开。

但赵嘉敏不认为他和陈昊芝的分歧是在于私人恩怨，而是对于“译言社区”的价值定位存在认识上的偏差：“分歧爆发的时刻，正好是我打算重点投入社区建设的时候。我觉得像译言这样的社区型网站，社区利益应该始终摆在最重要的位置。但陈昊芝那边，我觉得他不是这么想的。因为他不管是在对社区的回馈上，还是对社区贡献的态度上，都和我是另一种想法。包括他接手译言以后，曾经给译言的全体团队成员发了一封邮件，大意是译言今天所积累的所有内容，所有权都归译言。这让我很吃惊，一个做社区网站的负责人，怎么可以完全不顾社区用户的权利呢？所以我觉得和他最大的分歧，还是在于如何对待社区利益上。”

东西网：赵嘉敏的二次重构

离开译言网的赵嘉敏很快重振旗鼓，于2009年12月4日，和另外两个朋友出资创立了东西网，而核心团队正是当初译言的《卫报》项目团队。但赵嘉敏不认为此举是挖译言的墙角，他说：“首先，这支《卫报》团队因为项目合同到期，早在2009年11月就离开译言了；其次，在成立东西网之前，《卫报》团队曾经找译言谈判，希望能作为译言社区内的一支团队，承担译言的一些精品翻译项目，结果被拒绝了，所以我们也只能自谋出路了。”

不管怎么说，拥有了东西网这个平台，赵嘉敏至少可以自由地贯彻他的社区建设理念了。赵嘉敏指出，东西网不会成为译言的复制品，他说：“有趣的是，东西网成立的最初几个月，译言网一直在拷贝我们的模式。当然作为一种竞争策略，译言网此举也很正常。只不过等到2010年下半年，他们越来越难拷贝我们了，而东西网和译言网的气质差异也越来越大，基本上各走各的路了。”

赵嘉敏认为东西网无法再被译言网拷贝的原因，在于两者的用户群定位不同：“译言社区的受众面比较宽泛，而东西网定位在高端人群，走的是精品路线。” 走精品路线自然需要高端译者来支撑，据赵嘉敏介绍，目前东西网拥有400多个高端译者，其中活跃译者按周统计有40多个。在译者的管理方式上，同样采用的是“层级架构”模式，尤其是最具东西网特色的专栏产品，均由独立的社区翻译团队来负责。赵嘉敏认为这种专栏，没有层级架构的译言网尤其难以复制。当然，东西网的层级架构还是存在亟待解决的问题，尤其是当社区团队形成品牌后，难免会自立门户独立出去，不可避免会给东西网带来损失。赵嘉敏坦言，目前还没有找到一个好的方式，来建立土壤（东西网）和茁壮成长的树苗（品牌团队）之间更加长久的联系。

某种意义上说，东西网是赵嘉敏的又一次创业重构，但蕴含更加丰富的内容。现在的东西网，已经不像赵嘉敏当初经营译言网时那样一无所有：拥有11名员工的团队，每月净收入4万元，经过验证的众包模式、网络流量全球排名已经从创业时的0上升到18000左右……一切都在新的起点，亟待下一次的重构。展望未来，赵嘉敏表示东西网将着重解决两方面的问题：内容的产品化，以及社区的回馈模式。赵嘉敏透露，东西网将谋求与中信出版社一起，推进译者报酬制度的改革。

技术和世界的距离

回顾从当年的甲骨文工程师一步步走到今天的创业者，赵嘉敏认为从方法论的角度，技术工作和管理经营其实一脉相承，但技术背景的创业者由于其秉性容易犯三种错误：“首先，技术人员是非观太强，非黑即白，但在创业中需要学会变通；其次，技术人员总迷信‘技术至上’、‘技术改变世界’这类口号，事实上技术和世界之间，需要经历很多过程，如产品、用户，然后才能到达你的世界；第三，技术人员习惯在存在争议的时候才去沟通，而在创业时，往往等你发现需要争论的时候，一切都晚了——这是我在译言经历的教训，所以保持经常性的、开放性沟通是很有必要的。”

在结束本次采访时，赵嘉敏告诉记者：“尽管目前社会上的价值观很混乱，但我还是追求做一个具备起码道德准则的创业者。”

在中国的市场环境下做有道德感的商人，难道这也是运筹学法则下的可行之路？我们拭目以待。

手机安全领域的领航者

——记北京网秦天下科技有限公司CEO林宇

■ 记者 / 陈秋歌

细数众多成功的创业企业家，他们无不具有敏锐的商业嗅觉，能够快速把脉市场，发现并挖掘掩藏的商业先机，并成为新领域的开拓者与领导者，从而铸就事业上的成功与辉煌。早在2005年，当手机病毒还寥寥无几，业内人士仍在讨论手机病毒是不是伪命题时，林宇就十分看好手机安全服务这一行业，他确信在未来这将是一个朝阳产业。于是2005年林宇创立了北京网秦天下科技有限公司（下文简称为网秦），开始专注于手机安全服务。

创业方向是关键

林宇发现手机安全服务领域，出自于一个偶然的机会。2004年，他在北京邮电大学当老师。当时北京邮电大学网络与交换国家实验室与诺基亚已有十多年的合作，诺基亚科学家在一次访问该实验室时，提到手机上存在着安全威胁。那时手机病毒还很少，也只有一两款。凭借着敏感的商业嗅觉，林宇迅速捕捉到这一商机。他意识到手机安全将是一个新问题，其中蕴藏着新的机会和巨大的市场潜力。于是，2005年10月林宇离开了大学，并与两位校友创办了“网秦”公司，开始专注于手机安全服务领域。

网秦创立之初，手机安全市场还是一片空白，涉足该领域的企业寥寥无几，网秦算首家。巨大的市场开发潜力，让林宇对该市场的未来深信不疑。经过诸多探索与尝试，目前网秦已发展成国内手机安全服务领域的领先者。

创业的成功需要很多必要条件，但首要的条件是创业方向的正确选择。如何选择创业方向？林宇建议：“创业者在选择创业方向时应该注意四点。一是所选市场不要太大，但可以预见在未来五年内会有很大发展；二是对该市场有浓厚的兴趣。兴趣是动力之源，在兴趣的支撑下，才能克服千辛万难，把事业做大；三是创业团队在该市场领域有行业经验、技术等优势；最后一点是首选项目成本尽量低，因为大多创业者在创业之初都没有太多资金。”

风投是后盾

对于创业者来说，创业之初，面临的首要难题多是资金问题。网秦创立之初，只有10万元，可以说是杯水车薪。2005年至2007年两年间，由于资金的紧缺，网秦遭到了严重的考验，多次险些关门。但在网秦最艰难的时候，林宇依然看好手机安全服务领域。2006年8月，林宇的执着得到了一位天使投资者的认同，获得了100万的投资。这笔资金给网秦带来了希望，并帮助它度过了最艰难的时期。

经历了资金的这场危机，林宇意识到资金是公司坚实的后盾，必须主动寻找风险投资。当时他对风险投资领域完全不了解，于是便通过网络搜索，收集所有能收集到的风险投资商的联系方式，然后给他们群发邮件，介绍企业的技术和规划，并与国内外30多位知名风险投资负责人进行了面谈。也正是因为林宇对手机安全行业超前的敏感度及他对该行业的执着，

终于打动了金沙江、红杉资本，并获得了他们的投资，为网秦注入了新的血液。

林宇表示：“风险投资的进入带来的不仅仅是资金方面的变化，更重要的是改变了企业的视野半径，提高了公司管理层的独立经营能力及对市场、营销及战略做出正确判断的能力。从风险投资那里，我不仅学会了市场营销，更重要的是学会了企业经营之道——优秀的公司一定是在最需要花钱的地方放入最合适的资源。很多创业者最初并不太懂，但当公司到了一定规模的时候却非常重要。”

现在回想起当年艰苦的创业环境，林宇如此评价：一个艰苦的环境在锻炼团队的同时，也在挑战着团队的潜力。只有经受得住考验的团队，才能推出经得起考验的产品。

商业模式是核心

在网秦初涉手机安全领域时，林宇及创业伙伴就以全球的视野，把目光投向了海外市场。他们相信手机安全服务市场将是一个面向全球的市场。2005年10月，网秦发布了第一款双语版本的服务。半年后，经网秦后台数据统计，发现用户居然来自全球三十多个国家，这着实让整个团队为之振奋。为了把服务销售给全球的用户，在销售策略上网秦受网游点卡模式的启发，在eBay上做了首次尝试，通过卖点卡来提供服务。一两个月后，一位马来西亚的用户买了两张点卡——这是网秦卖出的第一份点卡。网秦的产品首次变成了商品，这对于林宇及整个团队来说，意义重大，是一次质的飞跃。

从一个Idea的萌发到产品原型的产生，再把原型转变成可供上百万用户使用的真正的产品，这是技术成熟的表现。如何把一个产品变成人们愿意付费购买的商品，再形成一个品牌，其中需采用的商业模式也是创业者不得不考虑的问题，而且是至关重要的。

林宇表示，从当时整个产业来看，传统的license商业模式肯定会受到挑战，最终用户会为服务付费，为差异化的用户体验付费，而不是为软件本身付费。目前网秦采用的是Free+Premium（免费+付费）的商业模式。

现在越来越多的互联网公司也都在采用这种模式。“互联网是一个早期产业，免费并非唯一常用模式，根据用户的付费意愿来定价也是合理的。对于有时间没钱的用户，他的付费意愿是零，那定价为零，即免费。这种免费的商业模式常常会把用户的注意力进行分类，然后卖给广告主。对于那些有钱没时间的用户，为他们提供好的服务、好的用户体验，他们还是愿意付费的。当然，用户付费要方便，同时让他能真正体会到享受到了有价值的、好的体验。”林宇如此认为。

林宇认为，只有经受过考验的团队才能推出经得起考验的产品

核心团队是支撑

网秦创立至今，已经历了五年多的风风雨雨，在手机安全领域也获得了丰硕的成果。

2010年9月，网秦被世界经济论坛评选为夏季达沃斯2011年“科技先锋” (Technology Pioneer 2011)，被《美国时代周刊》评价为“可以改变人们未来生活的十大创新企业之一”。10月，网秦入选全球四大权威会计事务所之一德勤所评选的“2010德勤高科技、高成长中国50强”TOP10。

谈到影响网秦成功发展的因素时，林宇认为最重要的因素是好的核心团队。技术产品、市场渠道、资本、社会资源及管理能力是公司创业成功的必要条件，而这些都离不开核心团队的支撑与达成。好的核心团队成员之间，在性格和能力上应该是互补的，并具备使公司成功的全方位的综合素质。在瞬息万变的市场环境中，好的核心团队可以对公司的方向和策略做出及时调整，使公司获得不断持续的发展并不断开拓创新的领域。另外在投资者看来，团队的综合能力也是很重要的考察因素。

网秦初创时期，只有林宇和两个创业合作伙伴。他们都是技术出身，既不懂市场，也不懂运营。为了把网秦创办好，他们不得不考虑业务上的分工。林宇负责产品、市场；另外两个伙伴一个负责运营、一个负责技术。三人组成了网秦的核心团队，在不断学习与摸索中，使网秦顺利成长。对于技术背景很好的创业者，林宇建议有两种选择，一种是寻找擅长产品、市场等其他领域的创业伙伴；另一种选择是做转型，并不断提升自己在其他领域的能力，当然这也需要背后有一个很好的团队来支撑。

技术壁垒是竞争力

2010年8月11日，国际知名的调研公司Frost & Sullivan发布的2010年上半年《中国手机安全市场白皮书》显示，现有国内手机安全市场份额，网秦占据 64.8%排名第一。网秦已由一个仅开发手机杀毒软件的公司发展成一个可以为用户提供手机防病毒/恶意软件、防骚扰、隐私保护、数据备份/恢复和数据管理等全方位服务的手机安全服务提供商。网秦能发展到今天的规模，林宇表示，除了得益于起步比较早之外，更重要的是开发了从客户端到安全平台全套的技术服务；同时，基于当前大的用户量及与产业链的深度合作，努力去构建了中国甚至全球最大的安全云知识库，这成为网秦发展过程中一个很重要的服务壁垒。

林宇说：“手机安全相对于传统安全，在技术上要做新的工作。因为手机环境的特殊性，比如内存和电池有限，这就要求手机上防病毒引擎要更加高效。此外安全服务，在某种意义上是一种垂直搜索的服务，这就要在云上构建一个很大的安全知识库。所有基于客户端的服务，实际都是由这个云上安全知识库来提供的。而构建云安全知识库，就需要有很大的用户群。随着服务规模的扩大、提供服务时间的延长，安全服务的技术壁垒也会不断提升。安全服务高技术壁垒的特点，将导致这个市场不会过于分散，公司也容易做大。所以对于创业公司，最好能找到用户可以频繁使用的普遍应用，同时这个方向还要具有核心壁垒性，只要坚持这样的创业方向，最终就能成长为一个很大的公司。”

通过网秦“云查杀”可快速检测/清除Android手机病毒

结语

随着智能手机的迅猛增长，预计三年以后智能手机的数量将会超过PC。充满诱惑的移动市场，必定吸引大量优秀公司的进入，竞争也会变得异常激烈。对于创业公司，谁能提供更多、更好的免费服务，提供更多有价值的服务，谁就能更多的占有市场。同时公司在不同时期要有自己清晰的定位，确定好用户目标群。一个公司的定位越清晰，越容易成功。林宇表示，准确的互联网战略，清晰的差异化市场定位，未来的国际化战略，对创业公司至关重要。

将“活雷锋”搬到网上

——记百姓网CEO王建硕

■ 记者 / 张雪峰

王建硕首次产生创业的想法是在1998年，那时他还是上海交通大学的学生。经过几年的学习，王建硕觉得应该用已经学到的知识做些事情。通过对行业的审视，他发现当时国内的电子商务才刚见雏形，并且也没有切实可行的在线交易方案。于是，王建硕决定与搭档李晓光创办一家在线B2C网站——Hotales.net。

回顾往昔，王建硕表示：“Hotales.net由于概念过于超前，在几乎尝试了所有的B2C形式后经营都不太理想。不过它后来转型成为了营销公司，一直很成功，现在已经是华东地区举足轻重的一家。”分析Hotales.net 前期不成功的原因，王建硕说：“当时并没有长远的规划，只是觉得自己的技术能力可以做出这样的产品，并且隐约觉得这很可能是未来的发展趋势。”虽然王建硕认为大学期间的创业尝试并没有成功，但是创业的种子，无疑已经在他的心里生根发芽。

创业，往往不需要安逸

大学毕业后，王建硕走进了微软位于徐家汇美罗大厦里的GTSC（微软全球技术支持中心）办公室。在微软的七年里，他几乎经历了所有与技术有关的职位：从工程师到技术、业务咨询，再到团队领导。这些经历给他带来的不仅仅是诱人的薪资和熠熠生辉的光环，更重要的是让他得以接触到全球的前沿技术，了解技术精英的优秀思维方式，以及微软颇具特色的管理理念，使他有机会实现自己的新颖创意，并参与到公司的先进技术研发。微软这样一个国际软件界的巨头，给王建硕提供了充分展示个人才华的广阔空间。王建硕对于这段经历曾这样评价：“就好像一个人在大学里，七年读了七个不同的专业，自然会拥有普通职员无法比拟的开阔视野。”

在常人看来，如果王建硕一直在微软发展下去，前方无疑是一条稳定的康庄大道。可对于内心深处萌动着创业种子的王建硕本人来说，这样的职业历程似乎缺了点什么。尽管王建硕在微软担任过许多不同的领导职位，也曾经建立过团队，但毕竟和自己创业有着极大的不同。

王建硕说：“在微软，只要产品完美，其他因素都是‘浮云’，任何一个产品都可以不惜一切代价将它做到完美，可以调动很多技术人员，将产品改了又改、做了又做，直到所有的瑕疵都被抹平，才会将其发布。而自己创业则不然，由于经济实力限制，不能无限地拖延产品的工期，更不能对一个产品投入不计成本的人力资源。另外，在大公司里，无论大事小事都要通过Email进行沟通；而在小公司里，把相关人员叫到一起商量事情则显得更加高效。”

所以，王建硕认为常年在微软这样的大公司工作，尽管会提高人的技术水平，但不利于保持创业必须的热情和技能。他说：“大公司有着相对优越的环境，许多事情都不需要个人考虑，可以只专注于技术本身。并且，在大公司里会使人产生错觉，认为自己很有能力。例如，在微软要搞一个主题演讲，通常会有上千人去参加，各个方面都有相关的支持，就连PPT都有专人来制作、整理。所有的配套‘服务’

在来华旅游的老外们心目中，王建硕是“博客活雷锋”

都非常到位。这样一来，整个人就会被公司的光环笼罩，使你错以为这是你自己的成就。但当个人离开公司之后就会发现，过去的那些客户、供应商都不复存在，你会发现这些并不是你个人的资源。总之，在大公司待久了，往往会使人贪图安逸，并抑制个人创业的想法。”

博客活雷锋

尽管在微软的职场生活让王建硕有些怅然若失，但“失之东隅，收之桑榆”，这段期间他不经意间培养的一个个人爱好——写博客，使得他拥有了一个与传统开发者截然不同的社会形象——博客活雷锋。而这段积累为他最终的创业奠定了坚实的基础。

一切缘起于王建硕曾经写的一篇关于浦东机场的英文文章，某次他不经意发现这篇文章竟然在Google排名第一。惊讶之余，他意识到老外在中国上海的生活肯定非常不方便，比如他们会迷路或者不知道哪里有西餐馆等。

王建硕说：“在美国，公共服务都会考虑到外国人的特殊需要，信息中心、语言服务点随处可见，租车、存钱、交水电费的时候都不会感到任何麻烦。”他觉得这是个很严重的缺陷，会给外国朋友带来许多不便之处。于是，王建硕从2002年开始有条理地搜集本土信息，并将这些信息通过自己的博客，提供给到上海来的外国朋友。但凡外国人需要他帮忙的话，便在给他的博客上留言，他基本上有求必应。这使得他在来中国旅游的老外群体中名声大噪，被誉为“博客活雷锋”。

有道是“苦心人，天不负”，王建硕通过博客，所表现出的对于本土信息的深入理解，引起了eBay的关注；同时，王建硕也发现eBay是业内少有的分类网站，双方一拍即合。2005年3月1日，eBay在全球80个城市推出了线上分类信息站点，客集齐作为其中之一在中国亮相，而掌舵者正是王建硕。随后，他秉承eBay“用各种本土化方式尽一切可能探试目标市场、攻破信息收集软肋”的理念，对其各项服务进行了深入的本土化探索。

让百姓上网更加方便

经过对国内互联网行业仔细的考察，王建硕发现了一个行业的缺口：有些人打算买一间自己的经济实力能够承受的二手房，有些人想将自己不再想住的房子卖掉；有些人想将旧车子卖掉，有些人想买一辆价格便宜的二手车。这些事情做起来都非常的麻烦，需要人们东奔西走，甚至走街串巷。看来想要买到一件二手的、令人满意的商品确实是一件耗费心力和体力的事情，所以如果能做一个面对个人、集各种二手商品交易于一身的界面交易平台，就可以很好地填补这一缺口。

鉴于上述体会，王建硕对于网站的分类进行了更加贴心的细化。比如，从提供给用户查询打折、兼职、陪驾、健身、家政、租房、快递、拼车、二手车等信息，到买卖转让电器、饰品、票券、化妆品、优惠券等个人生活信息。正如王

建硕所说：“我们就是为了让老百姓上网更方便！”尔后，王建硕逐渐觉得应该将网站的名称也进行本土化。2008年6月，作为eBay的全资子公司的客集齐，正式更名为“百姓网”这个更加本土化的名字。用服务对象——“老百姓”作为域名，也足见其独具匠心。

联系到王建硕当年专注于生活指南的英文博客，某种意义上说，百姓网是升级版的“博客活雷锋”，把活雷锋搬到了网站。

打造精英型轻公司

经过努力，王建硕已经将百姓网打造成了一家精英型的轻公司，全公司的的员工总数只有30人左右，仅占竞争对手公司人数的十分之一。30人的公司如何能战胜1000人的公司，这是许多人曾经对王建硕提出的疑问。对于管理，王建硕认为：“让大家保持一种自信是最重要的。我要让每一个团队成员坚信，只有最好的产品才可以赢得整个市场。”百姓网虽然是以交易为主题的分类网站，但是在王建硕的公司里并没有专门的销售团队，所有的宣传都是靠赢得用户的好评。王建硕说：“用户都是对百姓网进行口口相传的，我们相信只要做出最好的产品，产品本身会像被赋予魔力一样进行传播。”

结果百姓网在2008年被红鲱鱼杂志评为亚洲100强、中国10强。在全球ECG成员里面，百姓网也是利润率最高的公司之一。 王建硕表示：“互联网行业的核心是创造有价值的服务，多用服务器，少用人，这是互联网的精髓，也符合整个行业的发展规律。”对于是否会考虑扩大团队的规模，王建硕表现得很谨慎：“在团队方面我们和Groupon（团购网）不同，它的业务模式需要很大的线下团队。我们对人数来说没有具体要求，只是在招聘过程中门槛比较高。尽管我们业务的扩张需要很多人员，但是我们不会盲目地去扩招。人数增多一方面会增加生产力，另一方面也会增加团队沟通等各种负效应。”

目前，百姓网已经发展成为同类产品中交易速度最快、运营模式最健康的产品。并且，其服务已经遍及全国各大城市。王建硕强调：“在国内分类信息网站领域，我们不是收入最高的，但成本几乎为零，所以利润率一定是最高的。”

展望未来，“百姓网”已经把目光锁定在无线业务上，无线互联网是未来十年的金矿，王建硕也十分看好这个领域，他认为这是一个巨大的转折机会，而百姓网也在无线互联网上开发了相关应用，王建硕说：“这是个很受欢迎的应用。”为此，他们正在加紧开发手机版百姓网。

做自己擅长的事情

在企业平稳运作的过程中，王建硕已经总结出了一套完备的、适合创业公司的人才选拔标准。在微软，由于有“软件帝国”的光辉，各方面的人才都在源源不断地自己“找上门来”；而在创业公司里，则需要耐心地寻找。在这个过程中王建硕逐渐形成了自己对人才的评定标准。他认为，人才不一定要有许多工作经验，聪明才是最重要的。他说：“一个聪明的人，如果可以踏踏实实地工作，能耐得住寂寞，其他的问题都比较容易解决。”

从一名普通程序员到自己创业，王建硕告诫那些程序员出身且想自己创业的人：“一定要把握住自己的技术优势，不要急急忙忙地转向管理。目前的互联网行业，技术还是主导公司发展的重要因素。专注于自己擅长的技术领域，有利于企业的发展。管理、商务等其他自己比较陌生的业务，可以寻找一些擅长相关领域的合作伙伴。不要把精力放在自己不擅长的事情上。”

未来：分类将爆发

目前，国内SNS、团购网逐渐在发展壮大，占据了较大的市场份额。但是，王建硕表示百姓网会走自己的路，将分类交易服务作为唯一的发展方向，专注于把用户体验逐渐完善并将分类做到更多的中小城市，以及将分类继续细化。

对于分类网站的未来发展，王建硕认为：“直到现在，中国的分类信息行业仍处于起步阶段，还有很长的路要走。一是表现在用户的使用习惯上，在网上购买和出售二手物品的习惯尚未形成，全国范围内，用户交易的频率还不够；二是整个分类市场的规模还有待进一步扩大。未来五年分类网站将会有一个大的爆发。”

改变中国IT教育，从培训做起

——祝成科技创始人李建忠专访

■ 记者 / 木易

谈到大学里的计算机教育，周围的技术开发者鲜有不抱怨声讨的，和大多数人只是止于声讨不同，祝成科技创始人李建忠在大学时就认真考虑过现存教育体制的弊端，而工作后的经历包括一些图书写译、技术讲座的经历，使他对当前IT教育和业界实践之间的严重脱节更是有深刻的感受。于是他决定全身投入IT培训创业，所成立的祝成科技以重视“基础教育”为特色，在业界有不错的口碑。

《程序员》记者特别专访了李建忠，了解他的创业历程以及对国内IT培训市场的看法。

记者：您第一次萌生创业念头是什么时候？

李建忠：首次萌生创业念头源自我读本科时对大学教育的切肤之痛。那时我对计算机很感兴趣，但我发现在IT领域，整个大学教育体系是阻止学生进步的。我们那一拨学生中，计算机搞得好的，基本都是逃课生，并游离于大学教育之外。我发现自己的每一次进步基本都来自大学教育资源以外的因素（比如在网上听网友介绍一些技术网站，听技术同好介绍一些好书等）。

刚开始时对大学课堂的感受就是枯燥无趣，当时只是抱怨自己学校的老师不好，但是随着和很多其他高校的IT专业同学的接触，发现大家都有同样的感受。那时便开始慢慢觉察到可能是教育体制的问题了。记得当时也接触了一些资料，了解到五四时期蔡元培治下的北大，抗战时期的西南联大（大家知道中国近代以来的很多杰出人才都出自那两个时期），以及哈佛、斯坦福等世界名校。于是就在内心不断疑问“我们为什么不能产生那样的优秀大学？”也因此播下了后来创业的种子：自己将来是否可以做些什么来改变我们的大学教育。

不过，那时我只是一名大四的学生，虽然由于在很多科技竞赛中拿过奖，成为大学校园里的计算机明星学生，但我深知想以一己之力来办一所大学，无异于痴人说梦。

记者：后来促使你投入创业的原因是什么？

李建忠：真正将创业念头变成现实是在2004年（大学毕业两年之后）。那时候由于在IT技术杂志（包括《计算机世界》、《程序员》）上发表一些技术文章，也有译作出版，使得我有机会受邀到一些高校和培训机构做一些技术讲座，我在那里看到一双双渴望真知的眼睛——这让我不断回想起自己读大学时的那种状态。加上自己两年多的工作经历，我发现我们的IT高校教育体系和社会培训体系与IT产业一线严重脱节。

到了2004年下半年，我再也挡不住内心的驱使，那时的想法是：“我办不了一所大学，但是办一个培训机构还是有可能的。”于是便义无反顾地辞去了当时周围所有人都认为很好的工作，开始创办祝成科技。

说实话，我做辞职创业决定的时候，对今天大家耳熟能详的“商业模式”、“市场空间”等一无所知，我甚至连创业资金都没准备好（辞职三个月之后，我才发现自己需要的创业资金远远超过我的储蓄，才开始向亲友筹集原始资本）。

现在回想起来我对祝成科技的创业决定完全是冲动型的——对大学教育的切肤之痛，让我产生了“改变这种情况”的巨大创业冲动。不过六年下来，证明当初自己的冲动是正确的，我

李建忠表示，我的创业是基于一次勇敢的冲动

甚至庆幸自己当初有那样勇敢的创业冲动——如果按照常规的“商业模式”、“市场空间”等理性分析，这样的创业早就胎死腹中。

我确信祝成科技将是我第一次，也是最后一次创业。因为“改变中国教育”这个使命的空间足够大，足以使我拿出一生来奋斗。

记者：优秀的IT人才一般会选择留在研发型企业. 您是如何吸引优秀的人才加盟祝成培训的？

李建忠：我们既是一家IT培训公司，也是一家IT研发型企业。因此我们的人才分为几个部分：第一类是专家型的IT培训讲师；第二类是IT研发人员；第三类是业务及管理人员。

不得不承认，祝成科技作为创业公司在吸引人才方面有先天的劣势。但是另外一方面，基于共同的伟大愿景、富有活力的企业文化和高成长的激励机制，我们还是吸引了很多顶尖人才的加盟，比如前美国贝尔实验室科学家、著名IT管理技术专家薛君敖，台湾著名C++技术专家侯捷等。当然，相对于“改变中国教育”这一巨大使命来讲，我们现有的人才还远远不够，我们热忱欢迎各领域的优秀人才加盟祝成科技。

记者：祝成科技当前的主要业务面向企业客户而不是个人市场，这似乎和您的创业梦想不尽相符？

李建忠：我们在2004年刚开始创业瞄准的就是个人IT培训市场。但很快发现，在这个领域祝成科技最强的优势构不成企业的竞争力。那时个人IT培训市场最奏效的竞争力是“假、大、空”的宣传，大家比拼的是“谁的承诺更大、更假、更空”。只要能够通过“不择手段”的宣传，将学员忽悠进课堂，不管后续培训质量如何低劣，在商业上都算成功。作为程序员出身的我以及我的团队，无论从职业良知，还是创业理想来讲，都不允许我们这么做。发现这个痛苦的行业事实时，祝成科技正处于创业最困难时期，那时候企业的财务状况非常糟。幸运的是，我们发现问题后及时调整了方向——放弃“乌烟瘴气”的个人IT培训领域，转向面向国内IT企业的高端培训。这一定位的转向，加上我们在培训师资和课程方面的强大优势，公司的状况很快就发生了好转，赢得了包括许多世界500强在内的众多企业客户的赞誉和认可，企业发展取得了初步的成功。回头来看这次重大转型的成功，我想其核心原因在于：在个人IT培训领域绝大多数学员并非行业中的资深人士（很多是刚毕业的学生，或者IT行业之外的人士），因此对于IT培训企业没有很好的鉴别能力，对培训机构的选择非常不理性，“只看宣传和承诺，不看培训机构的实力，或者看不出培训机构真正的实力”，这就使得市场宣传得以成为这个行业的核心优势——这当然是一件挺悲哀的事情。

而在企业IT培训领域，客户都是行业中的资深从业者，选择培训企业非常理性，鉴别能力非常强，“假、大、空”的宣传和承诺在这里没有市场，实实在在的培训品质（培训师资、课程、服务）才是这里真正的竞争力。祝成科技在这方面的优势一下子就体现出来了，经过几年的努力便在IT行业内树立了很好的信誉和品牌。目前，我们有将近一半的企业客户都是老客户或者因口碑传播主动找上门来的。当然，个人IT培训领域仍然是一块重要的教育阵地——而且是我当初创业时最大的动机。虽然由于前述原因我们暂时性地放弃了这块阵地，但是长远来看我们对这一领域还是充满信心——毕竟我相信中国商业环境会随着互联网的发展而得到根本性的改善，只重营销而不重品质的企业会最终被人们抛弃。随着我们互联网战略的开展，我们会很快重回个人IT培训领域。我们将利用互联网这一巨大的平台，重构祝成科技的整个IT教育培训蓝图（目前这一工作正在进行中）。我坚信互联网会在教育培训领域释放出巨大的力量。我们内部的口号是“用互联网将大学的围墙推倒”。我相信祝成科技当初创业时的理想“办一所享誉百年的大学”会借助互联网的力量逐步实现。

记者：请谈谈您所提出的“用互联网重塑大学教育”的具体蓝图？和当前的在线培训模式有何不同？

李建忠：当前的在线培训只是一个简单的视频课件的播放，没能涵盖教育的各个环节。我们提出“用互联网来重塑大学教育”，是希望借助互联网技术平台来涵盖学

习的各个环节。换句话说，我们希望将整个学习过程搬到互联网上，而不是简单的将学习资源（视频）放在网上。从另外一个角度来看，目前互联网上主要承载的还是信息，而非知识。信息并不等同于知识。比如一个人想要学习C++语言，他/她通过Google搜索引擎、Baidu知道或者CSDN论坛等只能获取很多有关C++语言的信息。但是，这些都无法完整地帮助他/她完成C++语言的学习。我们技术团队正在研发一个基于互联网的知识平台，通过它，你可以完成自己的整个学习过程。随着这个技术平台的成熟，我们会逐步推出，我们相信它会改变许多人的学习方式，我们为这个愿景激动不已。

记者：以一个IT技术培训师的角度，您觉得中国目前的开发者主要有哪些问题？

李建忠：祝成科技创业的这几年，是中国IT业整个产业提升最快的几年，也是对IT开发人才需求最旺盛的时期。从一线培训教学实践来看，中国开发者最普遍的问题是基础理论功底薄弱。

中国传统的IT教育观点总是认为IT行业、特别是软件开发行业是一个实践出真知的行业。不管是大学教育，还是培训机构，都特别强调代码实战演练，但这实际上是一个很大的误区。

中国IT教育行业最大的问题是忽视基础理论，注重无价值的重复性实践。比如程序语言领域，堆栈内存模型是它的基础理论；再比如Web编程领域，HTTP协议是它的基础理论；桌面应用编程领域，操作系统、内存管理是它的基础理论……没有这些基础理论的支撑，编写再多的代码、再酷的程序，只是一种机械的重复，难有大的提升，而且会隐藏很深的漏洞。

我们在企业一线教学实践中，发现这个问题非常突出，所以会将相当时间放在“基础理论”的培训上，几年实践和学员反馈都表明，这种教学思路切中了我们IT教育的病根。

记者：在您的创业生涯里，有哪些经验令你体会深刻？

李建忠：在创业期间，令我印象最深刻和最值得珍视的经验是：第一，企业战略必须聚焦，抵制各种诱惑。第二，商业模式必须高度标准化。

企业战略是解决一个企业“做什么”的问题；商业模式是解决一个企业“怎么做”的问题。

在企业战略上，创业企业最容易犯的错误就是“满眼都是机会”，这肯定导致精力分散，不能做专、做精、做深。就像打井，一周七天，每天打一口井，可能都打不出水来。但是一周七天只围绕一口井打，出水的概率很高。我见过很多创业企业都死在了“战略不聚焦”上。

而对于商业模式，它只有实现了标准化，才具有很高的扩展性，企业才能快速做大。比如，中国酒店行业的老大不是某个顶尖的五星级酒店，而是汉庭、如家这样高度标准化的企业。再比如，中国餐饮业的老大也不是某个大饭店，而是麦当劳、肯德基这样有纯熟标准化的连锁企业。

记者：对于技术背景的创业者，您有哪些建议？

李建忠：技术背景的创业者，最容易犯两类错误：第一，容易将技术方向当作创业使命；第二，容易将技术人群当作客户。

创业使命首先是“解决人们的问题”，而不是“创造一个很酷的技术”。这一点技术人员刚开始往往容易犯错——我也在这方面犯过错误，所幸很快清醒过来。比如爱迪生发明电灯是为了给人们带来光明，而不是为电灯里面的技术而发明。再比如Bill Gates领导微软开发Windows操作系统，是因为GUI驱动的操作系统会将计算机的潜力释放给普罗大众，而不是因为Windows操作系统里面有很酷的技术。

一个企业必须有清晰的客户定位。技术人员由于交往人群比较窄，往往对“技术人员自己的需求”比较清晰，所以技术人员创业非常容易将技术人员这个群体假想为客户。这样显然大大窄化了客户空间。

当然技术背景的创业者也有很多优势，比如“学习速度快”、“专注产品和技术，而不是人际关系”，这些都是一家优秀企业基业常青不可或缺的基因。

提高“非技术性”能力是关键

——记 Trunk.ly 联合创始人董洵

■ 记者 / 付江

编者按：本文的主人公董洵是微软2004年评出的全球56个架构师MVP中唯一的中国人，在知名外企工作多年，此后经历了多次的创业历程，有苦有乐。他兴趣广泛，除了历史和艺术，还喜欢品南方茶、看书、听各式音乐以及运动。他还在继续创业，最近的项目是一款新型的社会化书签服务Trunk.ly。关于创业，他给记者留下印象最深的一句话就是，创业可能会把一个人在十年里受到的苦和误解，以及对未来的恐惧压缩在两到三年里承受。当然，你的成长速率可能也是加倍的。

去蹭计算机系的课吧

董洵与计算机的渊源可从吉林工业大学（现在的吉林大学）时开始，他当时的专业是机械工程。在大学之前，董洵其实并没有真正接触过计算机，第一次正式与计算机的亲密接触是在大学二年级的时候：“当时就特别喜欢，可以说是一见钟情。”从那以后，他开始在大学里“跳课”，每天去蹭软件学院的课：“当时学校上机费用挺贵的，45分钟要8毛钱，而在学校食堂里吃一顿带荤菜的饭也才1块多钱。”

好在当时不少计算机系的同学并不是真正喜欢这个专业，也不喜欢做课程设计和实习实践，于是董洵就帮他们代劳了。这样不仅上机费有人买单，还能把计算机软硬件专业知识都跟着自学了。

不仅如此，他还帮计算机系的同学上机写作业，按照机时收费，比如当时机房收1块2每小时的上机费，董洵则每个小时多收计算机系的同学两毛钱，帮他把作业完成，存放到软盘上，然后他再用剩下的钱请人帮自己写机械系的作业。由于当时计算机系的作业收费贵，机械系的便宜，行情好的时候，董洵还能剩下钱买软盘用（当时安装中文版Windows 3.11需要13张三寸盘）。董洵在大学里的很长一段时间内，都在乐此不疲着。

后话是，董洵在第一、第二学年的时候成绩还是机械系里前三名，到后两年基本上就是混及格的分数了。

董洵正式做软件，是从毕业前跟着学校博士生做项目开始的。离开学校来到北京后，董洵在上地的研华做Windows和Linux驱动程序开发。由于研华是一家硬件公司，而他更偏爱软件，所以在研华待了两年后又去了奥博杰天。

是不是该出来自己干了

奥博杰天是一家做高端外包的公司，从架构设计到开发测试全做。董洵在这一待就是近五年，前两年主要帮美国的一些创业公司做外包，最后一年多基本上是带着几个海归的博士做一些跟公司应用相关的科研。

董洵的创业心得：倒着走路的感觉

2004—2006年，因为工作需要，董洵接触到了很多美国的创业公司，自己也经常到美国这些创业公司待上1~2个月，一起做项目。他亲眼见证并参与了一些创业公司从诞生到快速成长再到被收购的过程。现在回顾起来，“创业的萌芽大概那时就已经在心底萌发了吧。”他笑言。

相信很多在大公司工作过的技术人员都会有类似的感觉，做到第7~8年，在公司越做越大的时候，其实更需要一些非技术性的东西“助力”才会有进一步向上的发展空间。这时有些人在团队管理上面可能用心比较多，就会开始向管理角色转型。董洵觉得自己还是更擅长技术和设计及产品运营，管人不是他的强项。然而当公司变得很大，当公司分工越来越细致，当研究和能力提升的空间越来越小的时候，董洵也开始考虑：是不是该出来自己干了？

其实在辞职出来创业之前，董洵已经在开始做一些自己的项目，例如“无废话新闻（类似于Reddit）。有一段时间，他每天早晨大概四点半起床，做（自己的项目）到七点半，然后吃早饭、坐公交去上班。当然遇上去国外出差，也经常会1~2个月都没时间弄自己的项目。不过，董洵也不是那种对工作随便混日子的人，同样很敬业，所以他越发觉得时间和精力不够用了。

在国外，只要一有时间，董洵就会到图书馆去看杂志，例如《连线》和《MIT技术评论》之类的；此外还会参加一些面向技术人员的聚会，这些都让他的眼界开阔不少，也发现了更多互联网上的创业机会。当创业的思绪和准备积累到一定程度的时候，他意识到：出来的时机到了。于是，董洵正式向公司递交了辞呈。

困难都是非技术性的

纵观国内，互联网创业比较成功的人都有自己的风格，王兴的风格是跟国外模式跟得很紧，陈一舟的风格是先按兵不动，看到哪个项目起来后，再利用自己的渠道全力出击。而董洵的创业产品需求则首先来源于自己，Trunk.ly便是如此。他说：“对于一款产品，首

先，我会问自己‘我需要吗？这个服务我会用吗？’，然后再去看，这项服务是不是除了我之外，还会有别人喜欢用这个服务。如果发现需要用到这个服务的人还真不少，而且是没有垄断的产品，我自己就会投资在技术上并把它做成原型推出去试验。如果发现越来越多人过来用，我就会向里面投入更多的资源，并继续把它做好。”看来，董洵很善于利用敏捷方法论的思想精髓。

对于开发人员自己创业，董洵第一个建议是：找一个有商务经验的、值得信赖的合伙人。例如如何去和媒体打交道、如何做市场、商务拓展以及谈判技巧，在这里面都有很多细节是开发者在创业的时候要特别注意的。

董洵现在做Trunk.ly的合伙人Tim Bull是2009年7月在澳大利亚悉尼一个为期两周的创业节上认识的，他是普华永道全球的四个总架构师之一。普华永道对这些高级员工有整套的商务培训，比如说怎么接受采访、如何做商务拓展。董洵举了个令他印象深刻的例子。当时在Trunk.ly上线后（恰逢雅虎宣布要关闭Del.icio.us），大量用户在寻找新的应用来转移书签数据，因此那段时间转移到Trunk.ly的用户数上升很快，服务器流量很大（他们临时还租用了Amazon的AWS云服务平台来分流）。那时刚好ReadWriteWeb的记者过来，采访完后直接就问：“现在有谁采访过你们了？”潜台词就是，如果有某个竞争媒体已经采访过你们的话我就立刻把稿子发出去，因为要抢新闻。

而当时董洵和Tim正在购买新的服务器准备做迁移，好几天没睡觉，特别累。他们想让记者把稿子押后两天再发布，等新服务器上线调试后再发，因为怕新闻发布出去后带过来的流量将把服务器冲垮。但这个话具体怎么说才合适，董洵实在拿捏不准。

Tim建议董洵，承诺给ReadWriteWeb首发权：“如果有别的媒体来采访，我们会在第一时间告诉你们，你们就可以立即把稿子发布出去。如果这几天没有其他媒体来采访，就麻烦你们，这个稿子押后两天再发。”这样就帮Trunk.ly赢得一些购买新服务器和调试的时间。后来事实是，大约一天半以后，又有一家大媒体来采访董洵，于是他们提前15分钟告诉ReadWriteWeb，这才把稿子发布出去了。

这件事情虽然不大，却代表了开发者创业之前意想不到的细节和技巧。这些能力在技术人的成长和工作中是没有机会得到训练的。但就是这些，例如谈判技巧、公关、商务拓展的经验和能力，在创业的时候尤为重要。

另外，如何在技术和投资Party上去和别人“搭讪”也是董洵刚开始创业时需要学习的内容。刚开始去参加Party的时候，面对几百人的会场，拥挤而吵闹，他甚至不知道该站在哪儿，更谈不上去考虑“如何与人交流，谈话内容说些什么”了。这时Tim又给他建议了几个细节：“没人跟你说话的时候，就看有没有人落单，或者新进来没有人跟他打招呼的情况下，你就主动去跟他说话，这个人就会很感激你。而如果碰到有几个人围着一个人交谈的时候，你想凑过去的话，需要站在一个合适角度，这样既不影响人家说话的人，也不影响听的人。如果你站得很偏的话，别人看不着你，如果你站得角度好，既不唐突，又能够让别人知道你在哪儿。”这些细节都是很重要的东西。诸如此类的非技术细节，董洵还遇到很多，他甚至笑言：“开发者创业遇到的技术问题都好办，困难基本都来自于非技术性的。”他建议开发人员创业的话，尤其要注意非技术性的综合能力的提升，当然最好就是找一个有商务经验的、值得信赖的合伙人。

朝着目标倒着走过去

当前，董洵的主要精力都在做一个Trunk.ly的新型社会化书签服务：用户在Twitter上分享的链接，在Delicious上收藏的内容，在Facebook上的信息，包括Google Reader、Instapaper上分享的链接，都可以通过Trunk.ly提供的书签服务进行“聚合”，并能提供对用户所关注者的链接的个性化搜索。

采访的最后，谈到创业心得，董洵给记者做了一个比喻：就像倒着走路的感觉。比如目的地是北京站，你要从你现在待的地方倒着走过去，你能知道的只能是你刚刚走的几步路和终点的大概方向，但究竟怎么走，是否走偏了，还需要在路上不断调整和夯实。

徐乐乐的日本创业记

■ 记者 / 杨东杰

编者按：在日本创业白手起家两年来，徐乐乐专注于教育应用领域，已成为App Store（日本区）在这一分类中最成功的独立开发团队。

有段时间，许多热爱英语的iPhone用户，都会相互推荐一款叫做iDaily的英语学习软件，这款软件因为外观精致、设计贴心，被推崇为“App Store上练习英语听力的最佳练习工具”。然而对于iDaily的创作者徐乐乐，感觉却很复杂。

因为据他统计，iDaily目前在全球的用户数已经接近20万，其中近5万是国内的盗版用户。

追随内心的声音创业

徐乐乐是位华人，但他的创业战场是在日本。iDaily只是他20多个成功应用里的一个。大学毕业后，徐乐乐来到日本，在一家中等规模的软件企业工作了6年，尽管职场顺利，从程序员一路做到高管，但他内心并不感到满足。2008年5月，他决定“听从内心的声音”，辞职创业。

数月后，iPhone 3G在日本上市，他尝试性地注册了开发者帐号，并开发了一个名叫“Best! 価格”的比价应用，结果上架后第三天就登上App Store（日本区）免费应用排行榜的榜首，许多用户给他发来了表达满意并希望改进的邮件，一下子让他感受到“自我实现”的满足，就这样，他头脑一热，选择了iPhone应用作为自己的创业方向。

在当时的日本市场，多数人并没有看到iPhone上市所带来的巨大机遇，诸如“iPhone在日本注定失败”之类的言论非常流行。但徐乐乐从2003年开始就是苹果产品的忠实用户，亲身经历了iPod在日本从非常小众发展到电车上人手一个，所以对iPhone的未来，他有着较常人更强的信心。2008年12月，徐乐乐注册了自己的公司——乐库科技，有意思的是两个合作伙伴都是“Best! 価格”的日本用户。

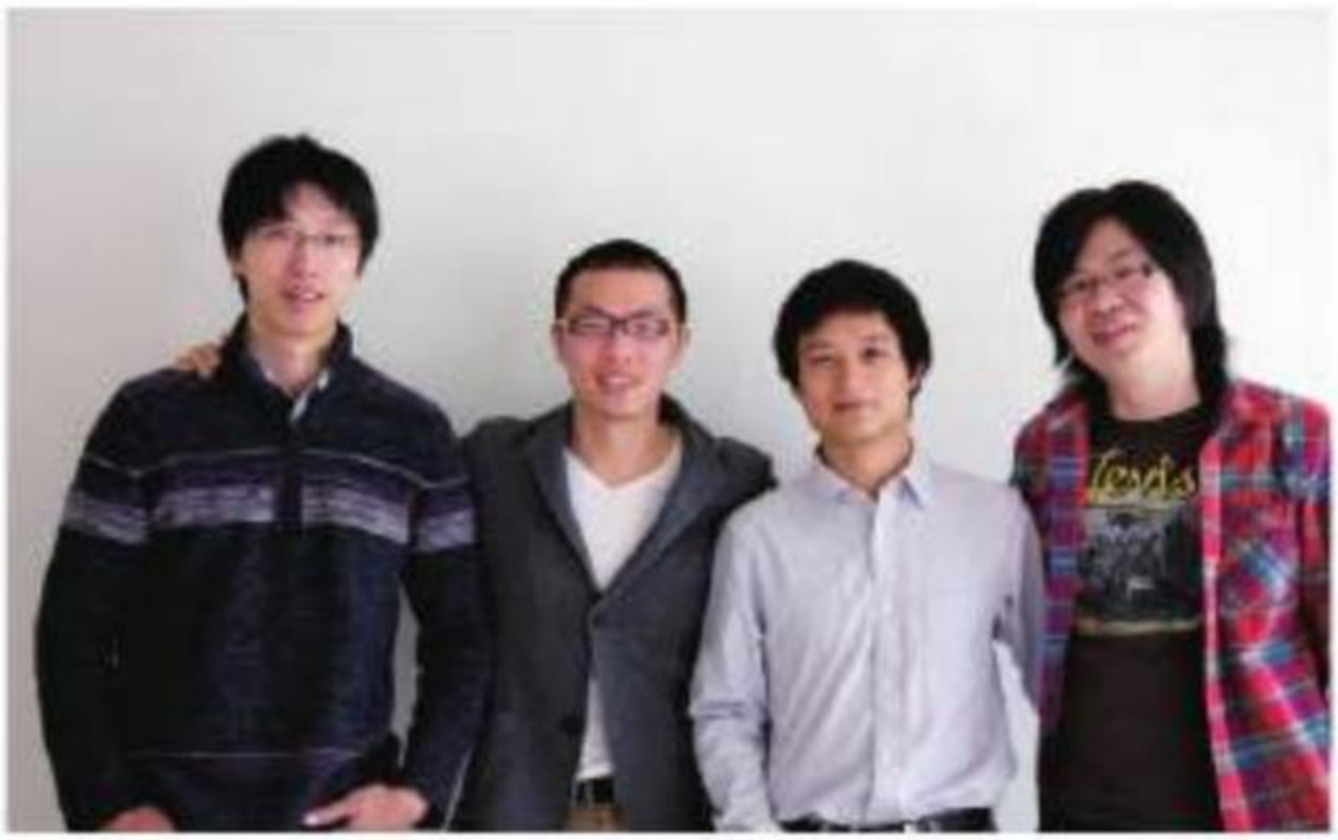
确立自己的专注领域——教育应用

比价应用之后，徐乐乐还推出了一款镜框应用，曾一度冲到App Store（日本区）总排行榜的第三名，但很快就被其他应用挤了下来。工具类应用竞争的激烈给徐乐乐留下了深刻的印象。此时，由于盈利的压力，他开始考虑要及早确定自己的优势领域。 他首先想到的是教育类应用：“日本众多的白领和学生是iPhone的主要用户，他们每天花费在电车上的时间超过了一小时，大多人是通过看书和玩游戏来消磨这段时间。”徐乐乐于是觉得，如果自己开发出英语听力学习类的手机应用，用更好的体验让用户能够随时随地学习，一定会大受欢迎。

碰巧的是，由于此时的徐乐乐已在iPhone应用开发者里闯出一定的名气，一家日本出版社主动找到他谈合作，随后他就推出了第一个教育应用——VOA英语听力，售价900日元，一上架就冲到App Store（日本区）教育类的第一名。

徐乐乐另一款教育类应用——“i发音”，由于优秀的人机交互体验，被苹果日本公司挑选成为最早登上iPhone电视广告的App之一。该广告在日本各大电视台的黄金时间滚动播出，给乐库科技带来了不少知名度，也让他们与苹果日本公司建立了良好的伙伴关系。

这两款教育应用的成功坚定了徐乐乐专注做教育类应用的信念。在日本，教育类应用一般都有正版的内容为依托，开发者需要和出版社在内容资源上进行合作。同工具类应用相比，由于有内容资源门槛，竞争并不激烈；而且教育类应用售价都在1200日元左右，比同样内容的书本便宜600日元，用户都乐于购买。对于徐乐乐这样没有任何外部资金支持的独立开发团队来说，选择教育类应用无疑更容易获得



乐库科技团队成员，左起依次为：浦力费（@baotuo）、徐乐乐（@xu_lele）、陈建明（@ashchan）、徐哲（@xuzhe）

稳定的收入，是明智之举。

成功来自机遇和实力

2009年年初，经原先合作的出版社介绍，徐乐乐成功地和日本最大的英文出版社ALC建立合作。两年里，徐乐乐的团队共推出了超过20款高质量的面向iPhone / iPad的产品，这些应用有多款拿下App Store（日本区）的收费榜第一、免费榜第一。在教育类软件中，更有四款软件被评入2009年和2010年iTunes日本区年度精品回顾榜。走进苹果公司位于东京银座区的日本旗舰店，几乎每一台iPhone/iPod展示机里都能看到乐库科技的产品。

对于自己的成功，徐乐乐多次表示，运气是很重要的因素。但徐乐乐能够在日本苹果应用市场特别是教育应用领域建立早期优势，所凭借的不仅是对需求的洞察和过人的眼力，徐乐乐本身深厚的技术实力也是重要因素。早在大学期间，徐乐乐对人机交互就非常着迷，他曾经是著名的0day组织成员，喜欢下载各种各样的软件来研究它们的人机交互是如何实现的。在iPhone应用开发中，他非常注重应用的简洁流畅，“i发音”之所以被选拔在iPhone电视广告中展示，也是因为其突出的用户体验。

在和日本ALC出版社的合作中，徐乐乐并没有采用流程化的批量开发模式，对于每个应用，都会选择那种能够展示出酷炫效果潜力的内容资源，投入针对性的全新设计开发理念，以保证每个应用都是精品。

通过做自己喜欢的事情赚钱

同每个创业团队一样，徐乐乐的公司也经历了许多困难：曾因为应用的Bug被用户们指责；因为App Store的限制无法及时地更新；因为海外服务器出现故障导致用户无法访问，于是为飞来的几十封投诉信而焦头烂额……尤其是2009年中，有两个月公司的收入还不够一个人一半的工资，当时徐乐乐感受到极大的压力，他说：“好在当时我正在做一个超有感觉的应用，一工作起来就全身心沉浸在开发中，也就忘了财务赤字这回事了。”徐乐乐笑称，这就是做自己喜爱的事情所带来的福利。

徐乐乐的创业团队每天的工作一般是这样的：上午10点多才到公司，工作到下午6点就一起开始玩星际争霸，然后就回家。当然工作忙起来就会加班到半夜。徐乐乐表示更看重团队成员的效能以及对工作的热爱。

多款成功产品推出之后，也有不少厂商找上门来谈应用外包合作，但都被徐乐乐拒绝了，原因是外包应用由客户需求主导，不是他喜欢的方式。也曾和VC谈过投资，但徐乐乐发现自己写不出VC喜欢的商业计划。他渴望盈利，但一定要以自己喜欢的方式来获得。

突破出版社合作模式 推出自有平台

2010年下半年，徐乐乐明显感到教育类应用的竞争更加激烈了，内容资源的优势逐步模糊，开发出的几款应用销量也不如以往，考虑到出版社对应用的推广效果不佳，徐乐乐不得不出让了部分利益来换得对推广的主导权。

除此之外，徐乐乐也在打造自有平台性应用iDaily，这是一款每日更新内容的英语综合学习软件。“这个产品的雏形在2008年就在我的计划中了，初步的设计和开发准备都已经完备，在2010年8月正式推出第一版，市场反馈很好，进入2010年iTunes日本区年度精品回顾榜单。”徐乐乐说。

徐乐乐认为教育应用的未来一定要拥抱“用户产生内容”和“社交元素”，iDaily存在着多种扩展方向，在下一个版本将有重大的升级，将从单一应用变成一个平台。在徐乐乐的计划中，一旦iDaily平台上拥有10万用户时，便引入外部资金和资源进行升级；至于对中国市场的拓展，目前也已经在他的计划中。

主持人

冯大辉，现任丁香园（http://www.dxy.cn）网站CTO。曾历任支付宝架构师、数据库团队负责人等职。

架构师接龙

张宴 vs. 岑文初

提问嘉宾

张宴，金山游戏运营技术中心技术经理。曾任金山游戏官网逍遥网架构师、新浪播客系统工程师。在Web、数据库、视频、负载均衡、搜索引擎、数据挖掘、网游运营等领域有丰富的架构设计、项目管理、程序开发和系统运维经验。

回答嘉宾

岑文初，2006年加入阿里巴巴，2007年初开始负责阿里软件平台架构设计，2007年底开始进入开放平台领域，2009年8月加入淘宝开放平台，现在是淘宝开放平台主架构师。自认没有特长，只是学习能力比较强。

张宴： 在项目的架构设计中，对于未来可能发生的需求变更，你是如何考虑的？如何应对？

岑文初： 需求变更可以分为业务性和非业务性两类。

对于业务性需求变更，思维方式应当按如下顺序进行：

第一，是否已经有类似功能，需要做些改进就可以满足需求；

第二，没有类似功能，是否可以抽取部分已有功能，再做部分封装即可实现；

第三，完全没有可以复用的内容，考虑一下后续可能的业务需求。

也许上面的内容比较虚，但业务一定是根据场景来做出实际判断，而这三点其实就是一个理念——不断优化业务代码，复用的思考会促进不断地合理化结构（因为大部分情况下，复用性越小的代码其结构本身存在耦合性过强的问题）。

非业务性需求变更主要是指由于系统自身应用场景发生变化（包括处理的数据量、业务规则复杂度等），而使得需要对现有系统做结构性调整。下面我举个例子。

开放平台的日志分析系统，在数据量不断增大和计算实时性要求增强的情况下，由单机多线程计算演变为多机分布式协作计算，从全量文件分析转变为增量基于数据流分析。但整体结构却没有发生太大变化，如图1所示。

最初，系统日访问量为六千万，数据分析报表每日出一次，分析器仅仅用于业务行为分析统计、单机每日拖取全量日志文件、多线程切割分析后合并。系统设计中有统计模型抽象层和简单的任务管理层：统计模型抽象层就是将传统的统计分析抽象成为对无结构化的数据做MapReduce计算，模型规则引擎可以根据配置直接分析无结构化定义的日志数据；任务管理层在第一阶段只是负责任务多线程内分配和管理。

当日志量每日达到两亿，分析器每日分析数据涵盖了系统、ISV、服务提供方，业务多角度分析和数据挖掘、单机I/O及CPU就成为瓶颈。因此，首先修改任务管理层，将原来多线程的任务管理和分配，扩展为支持多机任务管理和分配（基于TCP层的数据交互协议）。数据通信层理所当然地成为承载多机协作的基础层，因为本身没有太多的复杂流程在里面，就直接实现NIO的通信层。这次需求变更的代价就是扩展了任务分配协议以支持多机协作工作，本地的数据合并转变为通信传输后的本地数据合并，但对上层业务分析引擎没有任何影响。

当日志量每日达到八亿，分析器已经应用于整个系统的监控和告警及业务趋势分析等各方面，全量每日分析满足不了业务需求，于是由每日全量分析改变为实时增量分析。这次需要修改任务管理层：首先，扩展了Slave的数据源获取方式，除了支持文件获取，还支持HTTP数据流的获取；其次，扩展了Master对于任务周期的管理，任务列表可以被周期性重置，同时可以周期性导出增量数据，因此实时分析可以通过使用较短周期（分钟级别）的任务列表管理、任务分配、结果合并和增量导出，实现实时的大数据量分析。这次需求变更仅仅

图1 开放平台日志分析系统的整体结构

是对任务管理的周期做了更灵活的扩展，同时在Slave上做了多数据源数据的获取，对于本身的任务管理、合并、计算都没有任何影响。

总之，对于非业务性的需求改变，需要能够将业务性设计和系统设计隔离开来，同时明晰系统设计边界，支持外部接口的可扩展，这样就能够支撑由于系统应用场景的变化带来的系统架构的变化。

张宴： 在开放平台中，如何防范对于API应用接口的安全入侵、拒绝服务攻击？如何应对突发的访问请求激增情况？

岑文初： API的安全问题主要涉及两个方面：用户隐私安全和服务自身安全。

用户隐私安全，主要通过对应用的身份认证及用户主动授权两方面来保证。当然现在很多开放平台对于授权的时效性放得还比较松，这样会导致用户一次授权而被应用无限制使用的风险（用户往往找不到平台解除授权的地方）。淘宝开放平台，对应用做了级别划分和类型划分：首先在服务访问范围上，不同级别和类型的应用可访问的服务范围就被区别开来，因此高风险性的数据类服务只对Hosting的应用合作伙伴开放，不同类型的应用访问的服务也有所差别；其次，用户授权也根据应用类型的不同及服务本身的安全级别会有不同的授权时效性及授权提示，使得不同应用的操作用户数据会限制在不同时长内；最后，还提供了用户主动查询应用访问数据的情况及解绑授权的入口，为用户提供主动解除风险的机会。

服务自身安全，主要是应用自身主动和被动攻击（例如网站没有做数据缓存和防爬虫的功能，在爬虫爬取网页的同时，导致大量的无用服务请求）。当前开放平台主要通过两种模式多种维度来防止服务激增的问题。一种模式是主动在业务流程处理中按照简单规则计算并做访问流量控制，例如通过利用集中式和本地Cache作为计数器根据简单规则来屏蔽过量访问。另一种模式是通过记录日志，交由外部分析系统增量分析，外部分析器根据各种复杂关联性业务规则将分析后的决策推送给业务系统去做过量访问控制。这就涉及多种维度：IP、应用身份、用户身份、User Agent、访问的服务。这些维度可以简单地单个控制，也可以是组合控制。例如IP维度可以用在业务数据可能都是非法的情况下去屏蔽，由于规则很简单，可以直接放在主动模式下，而应用身份+用户身份比较复杂的控制，放在被动模式下，同时规则可以根据业务基线（不同日同时段、不同周同日的统计结果）的变化调整。从技术角度上来说，通过将业务线程池和容器线程池分开（类似于Servlet3容器线程生命周期不再固化与单一的方法），可以自己定义业务线程权重和处理流速来保证后端业务系统能力不会由于前端请求量激增时产生恶性循环。

张宴： 在开放平台中，面对不同背景、习惯的用户和开发者，如何能够让他们使用起来更便捷、易用？

岑文初： 在这方面，开放平台已经做和将要做以下事情。

第一，多语言SDK自动生成。淘宝当前有300多个API，而且服务还在不断增加，每次SDK的升级和发布都采用模板化方式自动生成，同时多语言的支持可以为开发者减小开发成本。这里就要谈到SDK的业务部分和非业务部分的抽象和隔离，更加有利于服务的增加和减少。

第二，服务数据化到服务流程化的转变。对于不同领域的开发者来说，淘宝的服务其实并不是都需要或者无差别的，例如淘江湖软件和商家管理软件就有很大差异，因此对于淘宝这样行业性认知要求比较高的服务，使用成本降低需要通过将服务由专家来做一定的流程化定制，避免使用的高门槛。

第三，提供Demo和Doc直接附加在SDK中，让开发者比较直接地看到使用方式，并且简单地运行一些具体的实例，增加感性认识。

第四，提供Wiki和社区，为服务升级和变更提供足够的信息传播渠道。

第五，提供线上运行环境和控制台，用直观体验降低门槛。

张宴： 在优化程序和降低系统资源消耗

上，有哪些经验可以分享？

岑文初： 第一，优化程序很大程度上是从全局而非局部去看，局部的优化可能导致全局的性能下降。例如，A系统和B系统是强依赖的，A的处理能力被提升后，到B的单位时间请求量会增加，而此时B服务能力达到瓶颈，单位事务处理时间就会增加，而A的处理能力增强并不一定会提高在A阶段的单位处理时间，因此整体事务处理时间增加，同时导致系统稳定性降低。因此优化需要关注更多的依赖，特别需要事先审视自己的系统是依赖流入还是流出。

第二，优化流程首先要切分流程的步骤，然后找到瓶颈点的步骤，而优化瓶颈点有两种方式，一种是直接优化减少瓶颈点资源消耗和处理时间；另一种是利用并行化方式，缩短资源生命周期，增加资源利用率从而减少对于资源的消耗，为瓶颈点的资源使用增加能力。

第三，合理利用外部计算资源，在可维护性和性能上找到平衡点。例如开放平台的结果返回处理通常情况下需要将对象根据规则定义转变为通用的格式。这部分工作可以交由开放平台集群来统一处理，也可以将规则处理后移到业务系统，前者的优势在于映射规则变动升级容易，而后者可以基于映射规则的成熟和无业务性，将计算工作分摊到各个业务集群上，因此计算外移是提高本系统性能的一种方式。

第四，可以将不同资源消耗型的应用部署在一起，合理利用资源（CPU消耗型、内存消耗型）。

第五，在做海量数据处理时，不要认为Java原生的东西都是无消耗的。尽量以最直接和简单的方式实现逻辑，用原生数据类型作为系统中间过程来处理内容（例如系统中就用long型做日期类处理）。

上面提到的也许看起来很抽象，但过于具体的细节优化其实遍及每个角落。优化最重要的就是找到全局的问题所在，同时在优化后是否会产生新的问题导致了性能反而不如优化前，以及系统稳定性也是优化所需要考虑的问题。

张宴： 在高并发应用中，Cache的作用不可忽视，在Cache的使用上，有哪些问题需要去注意？

岑文初： 第一，集中式缓存不是无代价的。从存储的数据量到交互次数上都需要去考虑如何降低成本。在海量并发请求的系统中，存储标识还是内容、多次交互还是一次交互，都会对系统产生很大影响。可以适当地使用多级缓存（remote + local），然后根据数据敏感度设置有效时间，简单处理数据失效问题。

第二，Cache不可用如何降级。对业务系统来说，一方面需要考虑Cache如何降级，也就是业务流程是否可以继续下去；另一方面如果Cache失效会从其他数据源获取数据，那么就需要考虑Cache的瞬间失效产生的峰值是否会直接击垮后端数据源。

第三，Cache如果采用数据源作为不命中的主动获取途径，那么需要防止无效的数据请求攻击透过Cache直接进入后端数据源。一般可以考虑用布隆算法来做增量白名单。

第四，注意使用好Cache提供的原子操作来避免并发带来的问题，例如add、replace、inc、dec等。

第五，需要去了解Cache的命中率和使用容量情况，不要为了技术而技术，需要更多的分析业务场景，最大限度地利用Cache的优势，同时减小存储消耗的代价。

张宴： 开源产品，很多时候可以用来借鉴设计思路、进行二次开发、引用部分类库，并应用到业务产品中。你是如何看待开源产品和业务产品之间的关系？

岑文初： 业务产品对于开源技术的选型需要慎重，同时切忌跟风。不必在意今天Facebook用了什么，明天Twitter用了什么，最重要的还是你自己的业务场景。如果可以用简单的技术实现，则优先考虑简单的技术借鉴部分开源设计的思想去做，同时量体裁衣简化系统设计。往往开源项目发展到一定规模会朝通用化方式发展，使用成本、附加功能、系统的松耦合，都会给中小型系统高速系统带来一定的负担。

使用开源项目最重要的是了解它解决的问题是什么、应用于什么场景、不同场景下的优势和劣势在什么地方、自身场景中使用的部分成熟度如何，最后做好足够的测试和验证，听来的总是虚的，因为业务场景不同，就算技术框架相同，结果也不同。

寻找机遇 创造未来 ——CSDN热门职位全新推荐

京东商城

[京东商城]诚聘手机开发工程师，软件开发工程师，欲试从速!

公告链接：http://www.360buy.com/intro/job.aspx

公司简介：

自2004年初正式涉足电子商务领域以来，京东商城一直保持高速成长，连续五年增长率均超过200%。京东商城始终坚持以纯电子商务模式运营，缩减中间环节，为消费者在第一时间提供优质的产品及满意的服务，目前，“京东价”已经成为国内3C销售领域的价格风向标。

京东商城在线销售商品包括家用电器、汽车用品；手机数码；电脑、软件、办公；家居、厨具、家装；服饰鞋帽；个护化妆；钟表首饰、礼品箱包；运动健康；母婴、玩具、乐器；食品饮料、保健品十大类逾10万种。目前京东商城拥有超过1000万的注册用户，日订单处理量突破7万单。现在，京东商城已经成为中国消费者选购3C产品的重要途径。

Zynga Beijing

长期招聘职位：

PHP软件开发工程师
Flash软件开发工程师
数据分析工程师
网站系统管理员

更多职位招聘信息和详细职位要求可登陆智联招聘或中华英才网站查询！

公司主页：www.zynga.com
地址：朝阳区远洋光华国际C座10层
加上：请以“职位+姓名+CSDN”形式命名的邮件，将简历发送至：BeijingJobs@zynga.com

欧特克软件（中国）有限公司

Autodesk 你一展身手的舞台，打造美好未来

你一展身手的舞台，打造美好未来加入我们的队伍，以一种全新的方式实现设计理念。无论是设计可持续性建筑还是开发混合动力汽车，Autodesk软件都可助你一臂之力。作为全球领先的二维与三维设计、工程和娱乐软件提供商，我们致力于帮助打造出更美好的未来。

让我们共同创造未来!
更多职位信息请浏览以下网站，www.autodesk.com/careers
或直接将简历发送至：acrd.hiring@autodesk.com

广州瀚信通信科技股份有限公司

岗位：Windows C#/C++开发工程师
工作地点：广州
职位描述：专注于移动通信信令类软件和系统平台产品开发
任职要求：
1.2年以上使用C#/C++开发Windows应用程序的经验；2.熟悉Windows应用程序开发，熟悉C#/.Net或C++，熟悉Visual Studio开发工具；3.了解ADO.NET数据库访问技术；4.熟悉SQL语言；5.具备英文阅读能力，学习能力强，思维敏捷；6.为人正直，工作勤奋主动，有责任心；7.具有良好的沟通能力和团队精神。

请以“职位+姓名+CSDN”作为应聘邮件标题，将简历发送至：yfhr.list@hantele.com
公司主页：www.hantele.com。

搜狐畅游

游戏研发部3D程序专员

工作职责：
1. 3D框架下各种功能模块和算法的实现；
2. 研究、开发和优化3D引擎。

职位要求：
1. 诚信正直，具备高度的责任心；2. 计算机相关专业本科及以上学历；3. 两年以上的3D引擎开发经验；4. 精通C++，面向对象编程，Windows编程；5. 熟悉3D图形学原理，熟悉3D几何；6. 精通DirectX/OpenGL，精通DX9；7. 具备良好的算法和数学基础，优秀的编码习惯和学习能力。

公司网址：http://www.changyou.com
E-mail：gamejob@job.cyou-inc.com

世纪乐知（北京）网络技术有限公司

杂志广告销售

岗位职责：
1. 负责《程序员》杂志广告销售，CTO专刊广告销售；
2. 与商务部其他同事一起协作承担商务部业绩指标；
3. 负责新客户开拓，及维护客户关系。

任职要求：
1. 从事过广告销售3年以上，平面媒体广告工作者优先；
2. 良好的商务意识，良好的职业道德及职业素养，积极上进；
3. 沟通能力强，热爱销售工作，敢于接受挑战，积极上进；
4. 有IT行业从业背景者优先考虑，英文良好者优先考虑。

电子邮箱： hrzhaopin@csdn.net

美国国家仪器上海研发中心

30多年来，美国国家仪器公司（NI）帮助测试、控制、设计领域的工程师与科学家解决了从设计、原型到发布过程中所遇到的种种挑战。通过现成可用的软件，如LabVIEW，以及高性价比的模块化硬件，NI帮助各领域的工程师不断创新，在缩短产品问世时间的同时有效降低开发成本。如今，NI为遍布全球各地的30,000家不同的客户提供多种应用选择。

NI上海软件研发工程师的发展方向

工业通讯：致力于将NI的软硬件平台应用到工业自动化与控制领域，结合目前流行的通讯协议，从事桌面与实时嵌入式系统上工业级软件的研发。

应用软件：紧密跟随业界前沿技术，深入到多个领域参与NI旗舰软件产品LabVIEW的开发。涉及领域包括： 图形化编译器、机器人及仿真、高速数据存取、工业仪器连接与控制、工业自动化控制和人机交互软件等。

招聘职位：高级软件工程师

主要职责：

1. 负责相关研发工作，设计、开发或强化产品，以及新产品定义
2. 紧跟技术前沿，并能将其应用到软件项目中
3. 实施和管理整个软件开发周期，包括需求分析、设计、编程和测试等
4. 撰写或审查面向用户的产品英文技术文档
5. 给下级成员提供指导和支持
6. 定期与其他国家同事沟通

职位要求：

1. 本科及以上学历，计算机、电子工程或相关专业
2. 5年以上软件研发工作经验，包括2年以上技术指导经验
3. 对于数据结构、面向对象程序分析和开发、软件架构和设计原理等有深入的理解和应用实践
4. 熟练使用C、C++等高级开发语言
5. 设计并实现过复杂的软件或功能模块
6. 有带领团队成功设计和代码检查经验
7. 英语口语流利，书面英语熟练

电子邮箱：China.jobs@ni.com 邮件主题请注明：应聘高级软件工程师_CSDN
更多职位信息，敬请访问：www.ni.com/china/career

NATIONAL INSTRUMENTS

人生就是在不断地做选择题，每一次选择，都会带来一点不同。而性格从根本上决定了我们每次面临选择时做出的决定。所谓性格决定命运，指的就是性格决定每一次做出的选择。所有的选择加起来，构成今天的不同，在别人看起来，这就是命运。

我的成长经历与心得

■ 文 / 张辉

计算机初体验

初中时，学校买了几台据说是苹果的电脑（其实就是一个键盘加显示器，主板内嵌在键盘下），于是我便有机会见识到计算机。但因为属于珍贵的教学仪器，一般人不能碰，所以这些机器的最终命运就在尘封的办公室里报废了。

1992年刚上高中，父母拿出积攒的3000元给我买了一台286，没有硬盘，只有五寸的软驱，绿色的单色显示器，内存也只有几百KB。就这样，我开始了第一段真正接触计算机的时光。因为没有硬盘，所以每次启动都需要先把启动盘插进去，系统引导完毕之后，再换一张应用程序的软盘进去。没有硬盘是个大问题，很多程序的使用都受到限制，结果当时电脑的最大用途就是玩一些简单的游戏，比如输入速度比赛游戏和俄罗斯方块。

虽然没写什么程序，但我还是对电脑情有独钟，因为能在上面实现很简单的操控感。

网讯安卓副总经理张辉

大学的收获

1996－2000年，我就读于西安交通大学计算机系。期间的最大的收获是认识了一些朋友，大家一起凑钱买电脑，一起（其中三个人）组队参加数学建模竞赛，获得了全国一等奖。后来又参加了两次美国大学生建模竞赛。记忆最深的一次是在春节期间参加的，大年三十那天，大家一起在学校招待所吃煎饺，看了几眼春节晚会，就开始连续72小时的封闭比赛。结束比赛，在清晨走出招待所的时候，有一种沉甸甸、晕沉沉的幸福感，这种幸福感，比获奖的时候要强烈很多。

十年后的今天，当初一起凑钱买电脑的其他三人，一个在深圳腾讯担任技术骨干，最早做企业IM；一个在上海，放弃清华博士学位，和同学一起创业做51mike.com，现在是国内一流的在线卡拉OK网站；一个在北京，华丽转身，掌管百亿的基金。虽然天南海北，但大家偶尔相逢，还是那么亲切。这也是大学四年留下的最宝贵的财富！我们都在各自的领域追求梦想、追求兴趣、追求热情所在。虽然无法去界定何谓成功，但我认为对于梦想和兴趣的追求，是我们这几个人还都生活得很惬意的原因所在。

这十年的职业生涯

工作的这十年可以用三个词来概括三个不同的阶段，那就是漂泊、稳定、重塑。

漂泊

1999年冬，大四找工作的我只投了一份简历，面试了一次，就定下去华为。我问面试官：“我适合做技术还是适合与人打交道？”当然，我并没有得到答案，但是却成为若干年后不断萦绕在心头、左右我工作选择的一个重要因素。

在华为差不多7个月后，因为个人原因婉谢领导的好意，选择离开，从深圳只身前往北京。

到了北京的第一份工作是在一家风险投资成立的科技公司，专注于做多功能的机顶盒产品。工作气氛不错，我也遇到了职场上第一位导师。当我还在深圳时，他就只凭一份简历和一个简短的电话认定我是他需要的人选。在这家公司一年多的时间，我从他那里学到了如何“通过代码证明自己的观点”。其实也很简单，任何时候，有猜想或者怀疑的时候，写一小段代码证明之，不要有太多假设和潜意识。所以后来我在带一些员工时，也是让他们用尽可能简单的事实或者代码证明他们自己的观点，而不是坐在那里空想。现在想起来，这倒与极限编程和互联网中常用的快速迭代反馈有一些相同之处。在这里，我完成了从一个学校人走向社会人的过程。

这家公司一直没有产品推向市场，更多还是Demo，这和我“写一段代码，供千万人使用”的理想相去甚远。后来经同事介绍，M公司北京研发中心招人做Linux手机，从此我开始了六年的M公司生涯。当时是2002年7月底，我唯一强烈的感觉就是漂泊不定的日子终于要结束了。

稳定

在M公司的六年，是一段非常快乐的时光，我亲身参与了一系列优秀产品的研发，从一个初级工程师最后成长为50人团队（包括20名实习生）的经理。

这期间最开心的事情莫过于看到一款产品从ID设计到编码、测试、发布，直到成为街机。那种感觉，充分满足了我“写一段代码，供千万人使用”的理想。据说当时最火的一款手机卖了300万部。当我们看到若干人手持透明盖的手机打电话、发短信时，一种幸福感油然而生。其实技术人员是很容易满足的族群，不是吗?

在后续的部门重组中，我们开玩笑说，这下子从地方军阀变成正规军了，钱多人多。鼎盛时期，全球产品团队多达千人，队伍遍布中国北京、美国、印度，开会时间都很难协调。现在回想起来，还是比较赞同“小团队”的做法。人，并非越多越好，尤其对于以软件工作为主的手机项目。因为团队太多，所以协调成本很高，有一些人员专门做“传声筒”，任何的决定都很难迅速推行。在队伍壮大的同时，我们也要面临重新定位，从以前的多面手，逐渐变成螺丝钉；从前的掌控感，逐渐变成迷失感。

在部门扩大的时候，我又得到一个机会，就是组建自己的团队，做部门内部的产品测试。测试本非我兴趣所在，但是想到有机会组建团队，按照自己喜欢的方式去架构团队，进行团队文化的塑造，和一群年轻有为的人一起共事，还是很开心。当队伍扩大到50人（包括实习生）、全职员工超过30人时，有一天晚上我对自己说：“从没有想过我竟然能做一个部门的经理，原本以为自己没有那种气场，没有领导风范。”我在整个学生生涯做得最多的其实就是“卫生委员”，说白了就是协调打扫卫生。小学有过唯一一次在众人面前发言，当时我非常紧张地读完预先准备的发言稿，然后走下台。台下有多少人，我浑然不知，只觉得紧张坏了。所以说工作是一个很好的媒介，通过与人共事，逐渐可以认识到冰山下面更隐秘的自己。

想增加“成功概率”，要做好四件事情：认清自己、认清大势、寻找一个好的环境、寻找“导师”。

测试部门日渐成熟，几个分管的团队领导也都各司其职，但我的失落感还是逐步增强。一方面因为自己兴趣并非在测试，整个部门流畅运转时自己反而难以找到更多的成就感；另一方面因为全球团队日益庞大，之前对于整个系统的掌控感消失殆尽。所以在工作六年后我萌生去意。

2007年在智能手机发展史上需要大书特书。首先是iPhone的诞生，使我们这些原本以为站在世界前沿的人第一次感觉到了“土”。而在iPhone宣布之后，GPhone谣言也满天飞，直到后来Android宣布推出。我隐隐感觉到一个新时代在诞生，只是没想到来得这么快、气势这么猛。其实我想所有员工都不免有同样的感受，若干年的光鲜似乎一朝就要蜕尽，再看我们正在做的一

2009年参观Google总部

些产品，着实很难有以往的自豪感可言。

在读完Visionmobile那篇分析Android商业模式的文章之后，“移动互联网”算是第一次真正深深地触动了我。我找到当时的领导说：“移动互联网是未来，我们应该抓住这个趋势。”从那之后，一定要做移动互联网的念头就深深植入我脑中。有时甚至想，距离移动互联网近一点就好，慢慢来。

2008年9月底，我离开工作六年的M公司，追随之前在M公司的领导，加盟了一家创业公司。他是在2007年加盟的这家公司，做的恰恰是和Android密切相关的开发。当时我隐约就有种感觉，苹果的iPhone和Google的Android，是未来若干年移动互联网领域最有影响力的平台。所以，要拥抱移动互联网，就得至少拥抱一家平台。苹果给第三方的公司空间小，只能做应用程序开发。所以，拥抱Android是明智的选择。

重塑

当在巨型公司逐渐迷失时，想找回“自我”和“掌控”的感觉就越发强烈。这也是我离开M公司，加盟一家创业公司的内在动因。

在随后的两年，在这家公司，我的工作是SDK的发布和开发者社区推广。这个与我在M公司中间两年的经历类似，所以做起来也是驾轻就熟。自己也借着这个机会，在各种大会以及各地奔波宣讲，辛苦之余，有不少难得的经历，甚至有机会去了中国台湾和美国硅谷。我经常给同事朋友讲，应该离客户近一些，离合作伙伴近一些。相比于之前的M公司，我在这家公司对外接触的机会要多了不少，这也是很好的收获。

凡事都有两面性，外界接触多了，了解得多了，忧患意识要比之前强很多。后来一次长途的自驾旅行，最终促使我做出了新的选择。

2010年7月，我做了一次横跨半个中国的长途自驾旅行，时间长达两周，距离超过5500公里。在远离北京、远离工作的地方回想起自己十年的工作生涯，感慨万千。从收入上，刚毕业时就很高，后续也是逐年快速增长，和同龄人比起来，应该没有太多遗憾之处；从职位上，在M公司的头五年里，连续升职四次，从一个入门级的初级工程师，迅速成长为部门经理，也还算不错；但想起未来十年、二十年的发展，甚至想到和这个社会一起步入老龄，就不免有些恐慌。十年以来在光鲜的岗位上积累起来的快乐感和幸福感是那么脆弱、那么不堪一击。

这种连续工作后的暂停，突然让我明白一个道理，在过往的十年，我和很多同龄人所追求的，其实都是“职业安全”，我们尽可能小心翼翼地在各种框架下遵循游戏规则，追求升职、加薪，过体面的生活，其实很可能只是一个泡泡。在社会各种产品、服务、资源价格上涨的今天，国内的用人成本也在不断提高，我们在过去十年看到的发达国家工作机会向国内的流动，现在有了新的变化，因为有劳动力价格相对更低的地区。十年前美国一名员工的成本，可以在国内养活4名同样优秀的员工，但十年后的今天，所谓白领的工资水平，正快速和美国的中产阶级接近。而这也意味着，我们之前在外企岗位上获得的“溢价”空间，在不断减少。好的工作机会，也在不断减少。之前引以为豪的快速发展的职业道路，可能会由于大环境的改变而减缓。

所以，长途旅行完，回到北京，我就积极寻找变化的机会，不是简单地换工作，而是寻求突破目前的“打工困境”。在将近一个月的时间里，我拿到了三个Offer，最后决定去最有挑战意义的一家创业公司，成为这家公司第一名员工。说实话，面对这三个几乎同时拿到的Offer，我也动摇过，不管内心再怎样憧憬创业，但是一碰到具体的数字、具体的金钱，再把这些数字换算

张辉的新团队

成那些自己想要的东西，比如一辆大众的小钢炮GTI，还是会小小动摇一下，没办法，自己只是一个普通人。

不过后来还是坚定地去了第一家给我Offer的创业公司。期间，有另外一家大公司的HR问我：“你不是之前就在创业公司吗？”我心想：“对不起，我还没有真正体会创业，那种诱惑无可抵挡。”

我算是进入这家创业公司的第一名员工，来这边已经三个多月，感觉过了很久同时也很快，因为这期间所做的事情、所做的决定，远远大于过往的任何一个90天。辛苦之余，也终于体会到什么是创业的感觉。

我们公司的模式很简单，就是把国内的Android程序带到国外市场，同时把国外的Android程序带到国内市场，我们专注做的角色是Publisher和Distributor。如果有什么例子可以做参考的话，最近因代理Angry Birds而声名鹊起的Chillingo应该是我们的一个偶像。之前和公司老总谈为什么要做这样一件事情的时候，我说自己并不是一个成功的程序员，起码没有成为自己心目中英雄那样的伟大程序员。但是从在M公司做SDK、在之前的公司做开发者社区推广，我就深深地意识到，虽然不能写出绝唱千古的代码，但是我有热情帮助那些能写出好代码的程序员成功。他们专注做技术，而我们则专注做渠道、做代理、做推广。我一直在举一个例子，就是在征友类电视节目中，有很多男女青年的职业都是网店店主，借助淘宝的平台谋生。我们为什么不能做出一个平台，打通从创意、设计、研发、发行、销售到消费的App价值链条。让有天赋、愿意自谋生路的开发人员、创意人才、设计人才在这个平台谋生、致富。10年后，我们相信会有一些眼里充满激情与渴望的年轻人的名片上印着独立开发者、设计师、创意师的头衔，可以像如今的网店店主一样自谋生路，惬意生活。所以，顺应形势，帮助别人去实现自己曾经的梦想——“程序致富”，岂不是一件快乐的事情？

我的心得

眼界高于能力，能力大于所得

在打工的环境下，在一个大型的、赢利良好的企业内，价值创造的链条被切成若干关联的环节，大部分基层员工都被局限在某个环节内，无法直接曝露在客户面前，更无从了解价值链条的细节。在这种情况下，有一个技巧可以分享，总结起来就是“眼界高于能力，能力大于所得”。

眼界指Vision，就是说虽然你可能是基层的工程师，但要有远大的理想，并且了解公司远景，最好找到合适的结合点。在做好本职工作的前提下，8小时之外坚持不断学习，提高水平，并及时和老板沟通心得。经常能站在一定高度和

老板进行沟通的人，他所能得到的机会要比整日默默无闻的人多得多，从而得到很多可以提升自己能力的实战机会，这就是所谓“眼界高于能力”。

“能力大于所得”指的是，如果你本身价值7000元月薪，那么6000元可能是一个更好的数字。因为所有的绩效考评都是人的主观判断，其中重要的一条就是横向比较。比如同样是贡献价值7000元的人，一个拿8000元，一个拿6000元。如果你是领导，你会给哪个人加薪？而如果这个薪资低一些的人，在8小时之外还非常努力，经常表现出更好的潜质，那又是什么结果？作为领导，你是不是还经常得考虑给这个性价比高、潜质更好、更加努力的同事多升职？如果你足够努力，总是能让“眼界高于能力，能力大于所得”的状态一直保持下去，那么持续的上升渠道就在你面前。而作为员工个人，往往又容易在这种状态下发挥出最大的潜能，从而形成正向反馈，加速提升。所以，在大企业的同学们应该特别注意这一点。

个人的价值与价格

我们周围的世界，充满各种言论、书籍、网站，教你如何成功、如何加薪、如何升职，但很少有人去探讨个人的价格与价值问题。价值本质上是自身能力创造出的财富，而价格则是雇主对于个人贡献的反馈。

基本的原理是这样，公司都是逐利机构，公司为客户提供产品或者服务，从客户那里拿到收入，然后除去各种费用，再分配给个人，剩下的就是公司赢利。这就是所谓的价值创造和利益分配的链条。

所以，对于创业公司而言，员工集体的薪资水平会影响公司盈亏平衡的时间早晚以及赢利水平的高低。我们在创业时，面试过一些优秀的人才，但是谈到最后一步，往往是因为薪资问题而没有谈拢。即使在北京这样一个高消费的城市里，月薪高过一定程度，对于个人的边际效应也快速减小。个人不会因为多几千元而更加开心或者离“财务自由”近一步，而公司却会因为相关的福利和税收系数付出更多。每一次创业公司被迫全员加薪，都会使公司距离自己平衡甚至IPO的目标更远一步，不异于饮鸩止渴。在这一个月内，我看到好几个事例，都是当事人在兴趣、能力甚至激情都具备的情况下，最后因为薪资而放弃这样一个创业机会。

其实对想创业以及正在创业的人才，都需要逐渐学习读懂财务报表，看看自己处于报表的哪一个区域。能从企业所有者和经营者的角度去看待待遇问题，是前期创业人员必须具备的素质。

我的建议

- 把微笑和与人为善作为每日习惯。最出名的例子来自于《程序员》杂志近期刊登的有关Android之父Andy Robin的故事。他只是在海边碰巧帮助了一名苹果公司的员工，后来就获得加盟苹果的机会，开始了一段又一段的传奇。
- 学会经常真心地称赞自己身边的同事、朋友，经常在力所能及的情况下帮助别人。人情就像储蓄，不要等到想用的时候才去存钱。
- 头十年的时候，你的价值可能只体现在拿到手的现金上，但往后的十年、二十年，你的价值更多的是人脉、眼光和社会影响力。寻找工作时，不要只盯着工资。只盯着工资，你的生活会越来越辛苦。
- 多出去旅行，同时多在旅途结识朋友。行万里路，永远胜过读万卷书。而“听君一席话，胜读十年书”也经常是正确的。
- 成功都需要一点运气，想要成功，你做的每一个决定都是增加这种成功的“概率”。想增加“成功概率”，有四件事情是最重要的：第一是认清自己；第二是认清大势；第三是寻找一个好的环境；第四也是最重要的就是要寻找“导师”，因为“导师”会帮你认清前三件事情。
- 帮别人创造价值在先，取得回报在后；帮别人成功在先，自己成功在后。你所能帮助的人越多，你距离自己的成功也就越近。想不通这件事情，你可能会一辈子辛苦。

作者简介 | 张辉

网讯安卓技术有限公司（Vtion Anzhuo)副总经理，负责Android应用软件全球发行代理工作和开发者社区关系。曾先后任职于华为、摩托罗拉、播思通讯。新浪微博：张辉forestsong。

■ 责任编辑：董世晓（dongsx@csdn.net）

SQA如何开展工作

■ 文 / 马越

作者以淘宝为例，详细介绍了SQA（软件质量保证）团队开展工作的正常流程，并最终构建出整个部门的流程框架。

面对一个陌生的环境，你必然会有很多疑问。例如，我在和公司外部的SQA交流或者面试的时候，谈到SQA的工作，他们问得最多的有那么几个问题，而如果我直接给出答案，对方会很茫然。

“你们是怎样检查流程的执行情况的？”——工具＋沟通、收集反馈＋数据。

“流程框架是什么？”——还没有统一的，一个部门一个样。

“你们收集哪些过程数据？”——研发过程上的数据，除了故障、各类Bug，很少有可以作为过程数据度量分析的，我们没有缺陷密度、开发生产率、代码规模、偏差等说法。

“你们SQA和开发的比例是多少？”——这个就更没有标准答案了，我只能说，一个SQA可能支持30人，也可能支持200人，也可能更多。

SQA是如何开展工作的？——这个是我下面要说的。

工作模式

我们的使命和愿景：打造高效、透明、稳定的研发环境，推动研发效率和线上生产质量的提升。

淘宝研发部门1000多人，对应不同的事业部，SQA团队分成产品线SQA和专题SQA两类。如图1所示。

- **产品线SQA**——从业务角度纵向支持，即按事业部配备SQA，从产品、开发、测试、发布、运维等各个环节做过程改进。
- **专题SQA**——从专题角度横向支持，

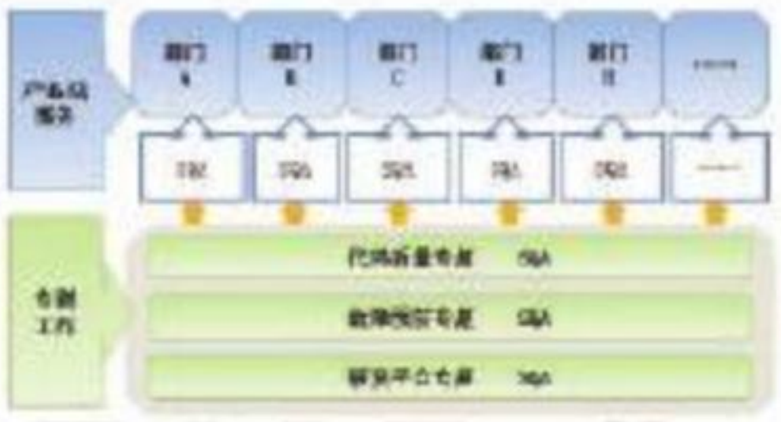


图1　淘宝SQA团队的划分

如故障专题、代码质量专题、流程平台工具专题。涉及各个业务部门时，需要产品线SQA的协助推动。

过程改进团队的定位

我们对过程改进团队的定位不是流程的监督者，而是服务者，需要发挥你的知识和经验，为客户部门提供服务。我们是站在产品研发环节之外，从第三方的角度看组织体系的运作，倾听每个环节人物的声音，通过组织里各个齿轮之间的摩擦和碰撞，寻找改进的机会，并且让各个齿轮之间更加润滑、更加高效和稳定。

说来简单，做起来却并非易事。对于在非互联网行业时间比较久的SQA来说，对定位的理解和心态的转变是很重要的一关，直接影响到提供给客户方案的效果和你自身的成就感。

认识客户

结合淘宝过程改进团队的使命以及SQA的定位，我们熟悉一个新客户团队的第一步，是去认识客户。

客户团队的背景

- **业务及团队目标**

去了解他们是做什么业务的，在整个淘宝网处于什么位置，即对用户、对公司的影响会是多大，他们近期业务目标是什么。这些方面可以找产品经理和技术人员了解。

- **近一年组织结构的演变过程**

组织结构调整在阿里巴巴体系里，是迅速且频繁的，而且每次调整也是有原因的，比如横向的职能部门变为纵向的垂直化管理、某些业务团队的合并或拆分。所谓“合久必分，分久必合”，组织随着日益壮大，不断尝试着更适应它的结构。了解这个演变过程，就帮助你熟悉演变的原因和历史的矛盾，更易于理解现状。有些问题可能当前无解，但随着组织结构调整可能就消失了，比流程上的改进要彻底。这些帮助我们判断什么是SQA当前能做的、可做的。

- **协作团队的评价**

正如相亲一样，亲朋友人对目标人物的评价会深刻地影响你。

我们支持的一般为开发团队，和他们接口的有产品、UED、Suffer、测试、PE这几个主要团队。这些团队与研发团队就像不同的齿轮，密切协作，他们对研发在资源分配、协作、项目管理、沟通等方面的反馈是很有价值的，比你去从各个平台报表里抠数据要迅速且准确得多。他们的抱怨既体现了研发团队的弱点，又反映出他们对研发的期望，而SQA通过与他们的交流也能识别出具备流程意识的“同盟”，当改进推动时，他们会鼎力相助。

这里，选择合适的人，正确客观地对待收集到的反馈很重要。

- **现有的流程制度**

每个部门或多或少都会有些流程规范，或者未成文的规定，可以去了解。虽然研发是按业务垂直划分的，提供基础支持的部门，如系统运维（OPS）、信息安全、DBA等也都是统一独立的，但这些部门发布的流程大多是要求各个研发团队统一执行的。SQA可以重点关注上线—运维环节、代码安全评审、DBA等方面的规范要求。

管理者

管理者有哪些人？这里主要包括产品和技术的负责人、各个Team的主管。

认识这些人！

他们就是你的重点客户，在做改进计划面临多个客户多种需求时，我们会根据客户优先级、改进任务优先级进行排序，确定工作的优先级。

认识他们不仅仅包括常规信息，还包括他们的性格、工作中的行事方式、在Team中的威信等。

SQA的工作就是和人打交道，而且，还从事着暴露出人家的毛病并让他们改正的工作，如何受到欢迎和认可？MBTI中把人的性格分为十六种类型，不同性格的人对待同一环境、同一问题可能作出截然不同的反应。通过了解对方的性格来更好地理解对方，进而在工作中建立信任和互信，寻求最合适的沟通和解决问题的方式。

- **管理风格**

由于管理者性格和团队所处的环境不同，管理理念和风格也不同。有亲和力为主的、有权威型为主的、有事无巨细型的、有充分授权型的，这些管理者对团队内部分工、团队倡导的氛围、对待流程规范的态度在很大程度上也是不一样的。

一些改进措施，比如是选择自上而下还是自下而上，需要根据环境、管理者的特点具体对待。

- **日常管理模式**

这里提的日常管理，主要指的是团队Leader之间是否有定期沟通汇报的形式、各个小Team内部成员管理的形式、绩效管理等。SQA在接触一个新的部门时，可以通过前面提到的一些沟通，有选择地参加一些例会，认识团队、让别人认识你、了解他们在做的事情、碰到的问题、团队氛围。

认识这些人，并让他们认识你。

如何抓住改进机会

在熟悉了客户的背景和人员之后，我们就要开始改进工作了。站在客户的角度来讲，我们的改进目标来源有：外部需求和内部需求。

外部需求，即客户团队外部的需求，如PE、OPS、DBA、信息安全、配管等部门需要推行某个流程，各个产品线的研发团队都要

执行。

内部需求，即客户团队内部的自身改进的需求。

如何识别改进需求？我们一般通过和客户关键人物沟通、对流程现状进行评估的方法进行需求调研。这种方式适用于在你加入之前，没有别的SQA同学支持这个部门，需要你亲自去调研。如果已经有SQA正在支持，则小组内先沟通一下现状和现有的改进计划，然后分工领任务。

思考客户的真实需求

我们一般会招聘在过程改进方面有丰富经验、对CMMI、敏捷开发等方面都具备系统知识的SQA。基于过程改进团队的定位，我们期望SQA能真正去解决客户的根本问题，避免流程过于理论化和教条化。

- **思考客户的真实需求，从思想上去贴近他们的真实感受。**
- **不局限于理论常规做法。**

在开发过程中，需要SQA更多地去思考客户的真正感受；对于改进方案，在特定的环境下，开发者的真实感受是什么、测试的真实感受是什么，以此类推，流程中各项活动涉及的其他角色在流程里的体验将是如何。站在他们的角度思考，而不是从理论或者规范的角度，也不是从审计、监督的角度。

但是，这个说起来容易，做好并不简单，比如说：

一个开发者觉得：我不喜欢束缚，干嘛要统计工时？我喜欢的就是coding和解决故障，多爽，写完代码丢出去就没事了……

这也是他“真实”的想法和需求。

所以，SQA需要分析我们的客户“群”，把客户细分之后，抓住重点，同时看到客户之间必然存在的矛盾，比如测试和开发之间天生的矛盾、产品经理和开发之间的矛盾，在他们各自“真实的需求”中找到正确的路，这是需要我们长期去修炼的。

此外，人的“真实需求”本身也是分层次的，举一个例子：我的“真实”需求是睡觉睡到自然醒，天天吃喝玩乐。可是我也需要钱，只好每天到点爬起来上班……

我们满足客户需求，层次不能太低，但是也不能太高。太理想化了，就没有意义了。什么是理想化？就是忽略了特定环境、人性的异同和团队成熟度，只追求理论上“应该”建立或改造的过程。

找准客户，细分好、再一一思考他们的需求。对每个改进活动，就能想清楚对他们有多少价值、ROI高不高。

确定改进的优先级

SQA的资源是有限的，本着对重点业务、重点项目优先保证资源的原则，在我们做改进规划和排计划的时候，也需要时刻考虑到任务的优先级。

- **从公司全局出发，能长远规划、影响范围比较大的改进。**
- **客户提出的需求或者问题，并且必须要解决的。**
- **对客户体验的提升有很大效果的改进。**

无论是过程的优化，还是流程工具的优化，或者是开发模式的改变，甚至是一个通知机制的改变，都离不开环境和人。下面我举一个例子。

在一个100多人的团队里，各个产品经理抱怨开发资源不足，需求总是延期，而且找不到一个整体负责项目的人，事情都靠产品经理去推进；开发觉得需求太多，就是人手不够，而且业务策略和需求都还经常变更，开发效率就被降低；测试觉得项目计划不靠谱，开发一延期，测试的时间就会被压缩……而这个团队还不断出故障。SQA仔细观察下来，觉得他们在项目管理上处于未开垦的状况，没有项目管理规范，开发负责人都凭借自己的能力和经验带项目，而这些负责人本身都很忙……从这个角度上看，他们迫切需要一套完善的项目管理规范。

但是，SQA发现当时并不是立即引入项目管理的最佳时机，主要是由于以下原因。

第一，这个团队上下还未深刻意识到项目管理的问题，认为最主要的是缺人。

第二，他们认为当前最急迫的是要避免故障，特别是发布过程中人为疏忽或者程序存在Bug、忽略对被调用模块的影响等因素而导致的故障。

第三，项目管理会涉及项目特性，不同

特性的管理流程可能会有不同，涉及项目经理的能力，项目经理需要在实践中提升能力，而这些投入比较大，短期之内却未必能看到效果。

如果不是开发团队有改进的意愿，那么在实施中会影响实施的效果，对今后的改进还会带来负面影响。虽然我们在多个场景下强调“流程Owner”的重要性，但还是会花时间通过SQA推动客户去意识到谁是真正的“Owner”——不是SQA，而是客户本身。

综合从上面几方面，我们采取的改进措施是抓一头一尾的方案，即从需求、上线这两个环节先进行控制。如图2所示。

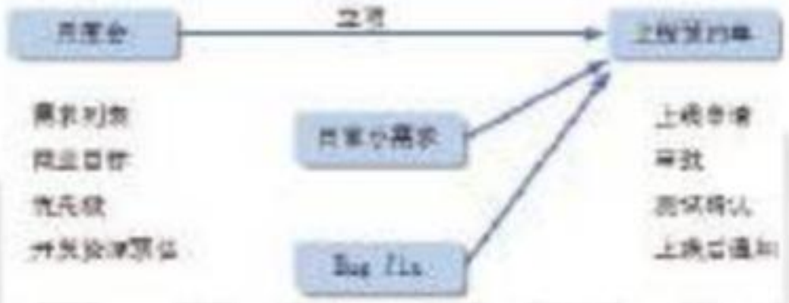


图2 从需求、上线环节进行控制

需求：让管理层看到当前可用资源的数据，以及产品经理准备要做的需求。以资源优先保证重点业务的重点项目为原则，对需求进行PK，PK的结果，就是未来1个月内确定有资源投入的项目。这个PK每月开一次。在组织第一次PK的过程中，把可用的开发资源与需求预估出来的所需的开发资源数据拿出来给大家看，客户反馈很强烈：“真是不盘点不知道咱们的项目有多大，不PK不清楚咱们的资源缺口有多大”，“让我们从整体到细节了解到了我们现在面临的挑战”……这种效果坚定了大家继续执行的信心。

上线：为了保证上线程序的质量和避免混乱及故障，SQA和开发、运维人员一起确定了上线预约的流程，提前发出预约通知，部门负责人审批，测试确认后方可上线，避免未测试就上线的情况。同时，用SharePoint建立支持这个流程的工具，能够自动发出各个阶段的通知，展现可视性。和改进之前相比较，对开发的体验来说就不是想发布就发布了，个别人会觉得不方便、审批影响效率。但是，部门负责人和运维人员很支持，测试更是非常支持，因此，这个流程也被坚定不移地执行下去。

这两项改进为项目管理打下了基础，让需求有序后，就是需要让项目顺畅运作了。在项目管理还没有推行的期间，SQA观察各团队项目的管理情况，未雨绸缪。比如：观察哪些人在项目管理上经验丰富，而且重视管理；哪些团队面临的挑战会比较大、最薄弱；项目类型上的区别；哪些人最迫切改变现状。虽然项目管理的优先级，在当时的情况是低的，但是在两个季度后，就变成了最高的优先级。而且，也是意料之中的，SQA所要做的，就是做好准备，抓住一切机会去推动，让客户团队里有影响的人去影响更多的人。最后，这个部门的流程框架得以建立。如图3所示。

图3 整个SQA部门的流程框架

总结

在改进计划中，建议SQA经常问自己下面几个问题。

造成这个现象的本质原因是什么？

这样做对谁有好处？能多赢吗？

这件事的ROI是什么？

当前的优先级是什么？

能回答出这几个问题，你就能大概明白何时能促进何种改进了。

作者简介 | 马越

资深过程改进工程师，先后在电子通信、网络安全、互联网行业从事软、硬件产品研发过程改进。2006年加入阿里巴巴，目前就职于淘宝技术保障部。

■ 责任编辑：董世晓（dongsx@csdn.net）

Marty Cagan谈软件产品管理系列

成功产品的规律及团队角色职责

■ 文 / Marty Cagan 译 / 兰蔚 刘雁

Marty Cagan是享有世界声誉的产品管理专家，曾经担任网景副总裁、eBay产品管理及设计高级副总裁。本文是他回顾自己二十多年来从事软件产品管理工作的总结和经验分享，谈到了成功产品遵循的十条规律以及产品团队的关键角色及其职责。

20世纪80年代中期我还年轻，在惠普担任程序员，参与开发一款备受瞩目的产品。当时人工智能风靡一时，能进入业内最优秀的公司，加入一支出类拔萃的团队（许多同事后来成为业界的中流砥柱），我感到非常荣幸。我们的任务难度不小：为低成本的通用工作站开发软件。当时市场上都是软硬件结合的专用产品，每个用户的花费超过十万美元——鲜有人负担得起。

我们辛勤工作了一年多，牺牲了无数个夜晚和周末。一路走来，我们为惠普增添了不少专利，开发出符合惠普严格品质要求的产品；我们把产品翻译成多种语言，实现国际化；我们还培训销售团队，向媒体进行展示，收到了良好的反馈。产品发布，我们以为万事俱备，开始庆贺。

但问题出现了：没人购买我们的产品。

这款产品在市场彻底失败了。是的，它的技术让人耳目一新，媒体反馈也不错，可是人们并不需要它。面对这个结果，团队成员很沮丧。但我们很快开始反省：谁有权决定开发什么产品？他们怎么决定的？他们怎么知道产品有没有用？

我们年轻的团队汲取了深刻教训，我相信很多团队也从失败中得到了同样的教训：如果开发的产品没有市场价值，无论开发团队多么优秀也无济于事。我们认识到仅仅做出产品并不够，还要确认产品是有价值的、可用的、可行的。

追溯产品失败的根源，我发现决定开发什么产品的人是“产品经理”，他们通常隶属于市场部门，负责定义我们开发的产品。同时，我还发现当时惠普并不擅长产品管理。不仅是惠普，大多数公司不谙此道，即便是今天有些公司依然如此。

我暗下决心，除非知道产品是用户需要的，否则我再也不会盲目投入精力。

此后二十多年，我有幸参与开发多款高科技产品。个人电脑兴起时在惠普工作；互联网技术爆发时，在网景/美国在线公司任平台及工具部门副总裁；电子商务风靡时，在eBay担任产品管理及设计高级副总裁。当然，并非所有产品都同样成功，但我可以自豪地说，没有一款是失败的。有几款产品广受欢迎，在全球拥有上千万的用户。

离开eBay不久，我接到一些产品公司的电话，对方希望改善开发产品的方式。与这些公司合作后，我发现他们的工作方式与优秀公司差异很大。我意识到普及一流产品理念的工作任重而道远。大多数公司都在使用过时且低效的方式定义和开发产品。同时我发现无论是学术机构（包括最好的商学院课程），还是那些因循守旧、无法自拔的公司（像我工作过的惠普），都对此无能为力。

我选择这个职业是想开发客户喜爱的、体现真正价值的产品。我发现产品主管都想打造让人眼前一亮的成功产品，可惜多数产品都缺乏创意、寿命短暂。因此，我希望通过自己的博客文章和著作《启示录：打造用户喜爱的产品》（*Inspired: How to Create Products Customers Love*）分享优秀企业的成功经验，让更多的产品赢得客户的厚爱。

我的文章涉及的内容

从担任网景高级产品经理开始，我的日常工作明确分为三块：人员、流程、产品。

Marty Cagan
过去20年，Marty Cagan作为负责定义和开发产品的高级经理人为多家一流企业工作过，包括惠普、网景通信、美国在线、eBay。他亲历了个人电脑、互联网、电子商务的起落沉浮，致力于通过写作、演讲、培训帮助客户打造富有创意的产品。为此，他撰写了《Inspired: How to Create Products Customers Love》，创办了硅谷产品集团公司（SVPG）。此前，他的最后一份工作是担任eBay产品管理及设计高级副总裁，负责规划全球电子商务网站的产品和服务。

人员指的是负责定义和开发产品的团队成员的角色和职责。

流程指的是探索、开发富有创意的产品时，反复应用的步骤和成功的实践经验。

产品指的是富有创意的产品具有的鲜明特性。

这三个部分是探索和开发用户喜爱产品必不可少的。项目都是由人完成的，流程则保证大家持续开发出用户喜爱的产品。我的文章将围绕这三个主题展开，内容多数来自一流公司的实践经验，有些得益于与业内精英交流的结果，剩下的则是我本人的工作经验。文章尽量均遵循以下三条标准：鼓励思考、与实际工作密切相关、切实可行。

希望我的文章能帮助你创造出成功的产品。同时我也非常希望听到读者分享自己的经验。欢迎访问我的博客（www.svpg.com）分享您的想法。

好产品靠设计

我从不认为富有创意的产品来自偶然，每款成功产品都遵循一定的规律，以下是我总结的十条规律。

- 产品经理的任务是探索产品的价值、可用性、可行性。
- 产品开发很重要也很困难，但用户体验设计通常比产品开发更重要更困难。
- 工程师不擅长用户体验设计，因为工程师脑子里想的是实现模型，而用户看重的是产品的概念模型。
- 用户体验设计就是交互设计、视觉设计（对硬件设备来说，则是工业设计）。
- 功能（产品需求）和用户体验设计密不可分。
- 产品创意必须尽早地、反复地接受目标用户的试用，以便获取有效的用户体验。
- 让用户试用高保真的产品原型，可以简单、方便、快捷地验证创意，获取真实的用户体验。
- 高保真产品原型是全体团队成员了解用户需求和用户体验最有效的途径。
- 产品经理的目标是在最短时间内把握复杂的市场/用户需求，确定产品的基本要求——价值、可用性、可行性。
- 一旦认定产品符合以上基本要求，它就是一个完整的概念，去掉任何因素，都不可能达到预期的结果。

产品团队的关键角色及其职责

产品是由团队的成员设计开发的。如何选择团队成员，界定工作责任，是产品成败的决定因素。许多产品团队在这方面显得因循守旧、捉襟见肘，他们会发现，我即将讨论的角色和职责与他们的做法大相径庭。并非所有公司都严格按我的方式设置职位、分配任务，但是大部分成功的公司是这样做的。这些角色是打造成功的软件产品不可或缺的。请注意，我所说的“软件产品”不仅包括提供给企业或消费者使用的软件，也包括互联网服务、电子消费产品，以及所有以软件为核心的设备。

产品经理

产品经理的主要职责分为两项：评估产品机会（Product Opportunity）；定义要开发的产品。产品创意的来源很多，包括公司高管的意见、用户的反馈、可用性测试的结果、产品团队和营销团队的点子、业内人士的分析等。应该有人严格审核这些创意，判断是否值得采纳。产品经理就是负责这项评估的人。许多公司借助市场需求文档（Market Requirements Document，MRD）来完成这项工作，但我更愿意使用一种简化后的方法，我称之为机会评估（Opportunity Assessment）。

确定有价值且符合公司发展要求的产品创意后，还需要探索产品的解决方案，包括基本的产品特征和功能、产品的用户体验、产品的发布标准。这些也属于产品经理的工作范畴，而且是产品经理的核心职责。有些公司借助产品需求文档（Product Requirements Document，PRD）来完成这项工作，也有人称之为产品说明文档或功能说明文档。同样，我主张采用简化的文档，围绕产品原型来展开这项工作。注意文档应该清晰地描述产品的功能和属性，避免讨论产品的实现方法。

用户体验设计师

实际上，用户体验设计团队由多种角色组成，稍后我会详细加以说明。这里只谈谈最

关键的角色——交互设计师（也称为信息架构师、用户界面设计师、用户体验架构师）。交互设计师负责深入理解目标用户（产品计划满足其需求的各种人物角色），设计有价值的、可用的目标功能、用户导航和产品使用流程。交互设计师与产品经理密切合作，将功能与设计相结合，满足用户需求。目标是确保产品同时具有可用性和吸引力（可用性指的是用户明白如何使用产品，吸引力指的是用户对产品的渴求程度）。

项目管理人员

产品经理完成产品定义后，开发团队承接项目，开始开发产品。项目管理的核心任务是制订计划和跟踪进度。项目管理工作常常由不同的角色承担,可能由专职的项目经理操刀，也可能由开发经理兼任（因为开发团队占有大部分项目资源），还可能由产品经理披挂上阵。这通常取决于公司的文化和项目的规模。规模较大的项目最好安排经验丰富的专职项目经理管理。

微软把负责制定产品说明文档和管理项目进度的人称为“项目”经理（Program Manager），由于这些可怜的人要同时应付多个项目，业界现在已经习惯用这个头衔称呼同时管理多个项目的管理人员。在微软，产品经理指的是那些负责的产品营销的人。虽然我不喜欢微软对这两个头衔的用法，但我认为他们定义产品的工作做得非常棒。

开发团队

工程师也称为产品开发人员或软件开发人员，负责开发产品，有些公司称之为IT（信息技术）团队，注意不要混淆这两个概念，区分的关键是看他们是为顾客开发软件，还是为公司内部（如人力资源部门）开发软件。IT团队通常指的是为内部员工提供技术支持的团队，而开发团队指的是为外部客户开发和维护产品的团队。

运维团队

互联网服务产品通常运行在服务器上，用户通过Web访问服务。运维团队负责保证服务正常运行。虽然有些公司将这项任务交给开发团队负责，但是运维工作需要一系列专业技能，很难由开发团队单独承担。

产品营销人员

产品营销团队负责对外发布、宣传产品，为拓展市场销售渠道、组织重点营销活动（如在线营销）、促进产品销售提供支持。有些公司让一个人同时负责产品管理（产品定义）和产品营销，这两项工作的技能要求相差很大，这样做实在不明智。

团队成员的构成比例

在产品团队里，产品经理、设计师和开发人员的人数存在一定的比例关系。为使开发人员集中精力开发有价值的软件产品，必须有相应人数的产品经理和设计师协助他们完成工作。

影响团队成员构成比例的因素包括待开发软件的类型、员工的工作经验和技能水平等。下面阐述的比例可供读者参考。

问：需要多少产品经理?

答：通常，每五到十位开发人员应该配备一名产品经理。

问：需要多少用户体验设计师?

答：一位交互设计师大约可以支持两位产品经理的工作，一位视觉设计师一般可以支持四位交互设计师的工作。

问：应该聘请专职的项目经理吗?

答：凡超过十名开发人员参与的重大项目，就应该配备专门的项目经理。此外，如果采用火车模型发布法（指以固定的周期持续发布产品，如果某项既定功能未完成，就挪到下个周期发布），必须为每次产品发布（通常这类产品由多个项目的组成）配备专职的项目经理。

本文节选自华中科技大学出版社《启示录：打造用户喜爱的产品》一书。该书从人员、流程、产品三个角度介绍了现代软件（互联网）产品管理的实践经验和理念。特此感谢华中科技大学出版社与Marty Cagan先生授权。

中国产品经理如何突破

■ 文 / 马博

通过对比分析国内外两家IT企业产品经理的岗位职责和任职要求，指出了国内产品经理的三大差距，并给出了中国产品经理的突破之法。

产品经理职位在中国发展非常迅速，还记得UCPM（中国产品经理联盟）成立之初，鲜有人知道这个职位，更不知道它具体做什么工作或者有什么价值。时过境迁，现在产品经理已经被企业普遍采用，而且其数量也在惊人地增加。但反观产品经理的职业领域，发展速度远远没有跟上企业需求的速度。这导致一个很大的问题，就是产品经理的职业能力普遍不足，没有起到一个产品经理真正的作用，更没有体现产品经理职位的价值。产品经理本来是为企业解困，现在却成为企业为之所困的职位。

造成这个问题的原因很多，但我们可以从产品经理职位本身出发来分析和探讨症结及其解决方法。产品经理职位起源于美国，在美国发展了80多年的时间，在职业化和企业基本层面都很成熟。那么我们就以国内外两家典型的软件企业为例，来分析国内产品经理的差距。通过对比分析可以更好地认识我们的产品经理还有哪些方面能够突破现状，继而提升自身的职业能力。

国内产品经理的职责和任职要求

先来看国内某软件企业对产品经理的岗位职责说明。

- 负责整理产品需求，定制产品架构（绘制使用流程图、制作高保真产品原型），完成详细的产品需求文档。
- 分析现有产品功能，跟踪行业最新发展趋势，完成具有独特见解、创意性的产品发展方案。
- 协调各部门沟通，推动项目进展。
- 一切以用户价值为出发点，深入了解、分析并持续挖掘用户需求。
- 关注竞争对手状况及新的商业赢利模式，对市场变化反应敏锐。

下面是该企业对这个岗位的任职要求。

- 能独立编写产品的详细流程、产品需求文档，有较强的创意设计概念。
- 较强的逻辑思维与系统分析能力，良好的数据分析能力，能积极思考行业动态变化及新事物，有效挖掘提炼形成新的产品概念。
- 了解互联网产品的开发基本流程，熟悉用户研究、可用性研究以及交互设计相关知识。
- 了解互联网行业相关赢利模式，具有商业意识。
- 了解各类互联网应用产品，有用户研究、产品策划相关实习经验者优先。
- 工作认真负责，态度端正，很好的沟通和协调能力，富有团队合作精神；想象力丰富，有激情。
- 主动提出自己的观点和意见，能够使用文字与言语流畅地表达自身思想。

好，我们对这家企业的产品经理的岗位职责和任职要求进行一个简单的分析。

一般来说，企业会按照期望的工作职责的重要性排序，那么，这家企业对产品经理岗位职责的描述说明它认为产品经理主要的工作就是完成PRD（产品需求文档）。

当然，完成PRD确实应该是产品经理要做的工作，但这是否是产品经理最为核心的工作呢？这里先留个伏笔，接下来会讲到。

此外，其他四项职责分别涉及：产品发展

方案的制订、部门沟通、用户需求管理、市场敏感性。

而任职要求基本上都是围绕产品的概念化、图纸化，以及一些具体的技能进行说明的，例如可用性（Usability）和交互设计（Interaction Designer或者UE），其他就是一些个人素养方面的能力要求了，例如沟通能力。

国外产品经理的职责和任职要求

下面我们来看看微软对其嵌入式产品经理都有什么样的职责和要求（为便于阅读，采用中英文对照的形式来说明）。如表1和表2所示。

表1　微软产品经理的岗位职责

工作职责	
英文	中文
Market research and analysis including sizing	市场大小的研究和分析。
Value propositions development and validation	价值主张的开发和确定。
SKU and pricing strategy, Defining roadmap	单品和价格战略，定义产品路线图。
Messaging and Positioning	沟通和定位。
Core content creation (e.g. sales tools, collateral and provide guidance to MARCOM)	核心内容创建（例如销售工具、collateral和提供营销传播指导）。
Analyst and reviewer engagement	分析和评估合约。
Strong analytical and cross-group collaboration skills to bring the organization to consensus on decisions	在统一组织决策共识上具有强大的分析和跨职能合作能力。
MSFT OEM field organization engagement as a subject matter expert	作为一个专家参与微软OEM领域的组织。

表2　微软产品经理的任职要求

职位要求	
英文	中文
6 years experience in product management and or product planning	6年的产品管理或者产品规划经验。
Passion for the rapidly evolving embedded device market	对快速发展的嵌入式设备市场有着强烈的爱好。
Strong written and oral communications kills	强大的撰写和口头交流能力。
Analytical problem-solving capability	分析解决问题的能力。
Ability to collaborate with diverse group in a cross functional organization	在跨职能组织中和不同团队合作的能力。
优先考虑条件	
Experience in embedded software/ products	有嵌入式软件/产品经验。
Experience in embedded device management	有嵌入式设备管理的经验。
MBA	MBA

其实看到这里，无需再做太多说明，大家也能看出这两家企业在产品经理方面的差异了。微软产品经理的主要工作是在市场细分、市场规模预测、产品定位和价值的设计与传递、价格策略的制定、路线图、销售支持、对外合作等方面。职位要求除了必要的行业经验外，主要就是强调沟通、分析解决问题和跨团队合作的能力。此外，如果是MBA的话，也是优先考虑的条件。

国内外产品经理的三大差距

通过上面两个案例可以看出，在微软，产品经理属于营销岗，而在国内，软件行业的产品经理还是被定位成策划或者设计岗（事实上，国内那家企业的产品经理确实属于策划岗）。

如果说这个结论——产品经理属于营销岗——是对国内其他行业说的（例如医药、消费品），那么他们会很欣然地接受。但如果是对软件行业，或者再延伸到整个IT领域，那么这个结论可能就有些不太让人容易接受了。

IT行业的产品经理怎么可能是属于营销岗的呢?

这就是我要谈的国内外企业在产品管理上第一个差距：**产品管理属于营销岗而非技术、策划或者运营岗**。

产品管理从诞生之日起，就被打上了深深的营销管理的烙印，这是产品管理发展的基础，即使是国内其他行业，也是这样认识的。但中国的IT行业在开始采用产品管理后，就把

这个岗位“创造性”地延伸到了技术岗，很多产品经理或者产品管理部门竟然都是隶属于研发中心或者是负责研发的VP下，也就是说，要么是作为研发部门的二级部门，要么就是和研发同属于技术层面的一个平级部门。

其实，每天看看这些产品经理在做什么也就印证了这点。需求、功能、UI、UE、原型、PRD、FRD，每天就是围绕这些在开展工作，当然，并不是说这些工作不应该去做，而是说这些工作，产品经理做得太多、太深了，而真正需要产品经理去做的，例如市场问题分析、产品战略制定、产品价值设计和定位、产品发展路线、产品营销策略、价格策略、渠道赢利评估等，则几乎一点都没有涉及。

你去问他们为什么不去做这个，他们会委屈地告诉你：公司没有告诉我应该去做这个啊。

没错，这就是中国的IT企业对产品管理的认识，既然你属于技术岗，就是为技术服务的，了解这些战略、规划、品牌、价格、渠道的知识又有什么用呢？

这就又引出了我们要谈的中外企业的第二个差距：**产品经理应该做些什么**。

IT行业的产品经理有一个错误认识，总认为自己的产品管理和其他行业是不同的。

这点在国外就形成了非常一致的认识，产品管理就是属于营销岗，在任何行业都是这样，无论你是快消行业（例如宝洁、联合利华）、医药行业（例如强生、辉瑞），还是汽车行业（例如福特、本田），或是IT行业（例如微软、Adobe），都是如此，区别仅仅是在于因为产品管理架构的不同而在工作的深度上有些差异罢了。

但是，具体要做的工作几乎都是一样的，并且他们也基本形成了一种普遍共识的产品管理工作流程和规范。虽然在具体的实践上有不同的流派，但是基本都包含战略、规划、战术三大活动，在这三大活动里，又有十一个阶段，每个阶段里包含不同的工作，大致有三十七项。

而国内呢，别说基于统一的认识去共同推动产品管理的发展了，自己连产品管理是什么还没搞清楚呢。

有些朋友可能会不满了，你说的那些我也知道啊，但是问题是，光知道有什么用，知道怎么去做才是关键的。

这就是我要谈的第三个差距：**国内缺乏对产品管理从业者正确、体系和职业化的引导**。

虽然国外同行在产品管理的认知、工作内容上的认知上有着一致性，但是具体到从业的个体上，一定会有比较大的差异。同样，国内的从业者也有这样的问题。但国外比国内领先的一点是，他们都是基于统一的认知思想、职业平台在开展着交流。

而国内虽然也有一些，但其中很多都是基于前面提到的片面的产品管理认识在进行交流，这不是越交流越偏、越交流越错吗？还指望什么提高呢？

这还算是好的，更有甚者，还有不少滥竽充数、偷换概念的。如果说片面认知的结果是原地踏步的话，那么这种情况就是彻头彻尾地让中国的产品管理倒退了。

好，通过对国内外产品管理的比较可以看出，我们与国外同行存在三个明显的差距。

认知上的差距：IT行业对产品管理的认知过于肤浅和片面，这直接决定了企业会如何认识和实施产品管理。

具体的影响会包括基于产品管理的业务体系架构，产品管理流程和规范的设计，产品管理人员的招聘、使用和发展以及产品管理体系价值的体现。

工作上的差距：正是因为有了认知上的差距，才造成了在工作职责定义上的差距，国外是基于产品管理的本质来定义产品经理的工作，而国内的IT行业则是基于产品设计、产品策划来定义产品经理的工作，虽说产品经理也会涉及一些设计和策划的工作，但那只是很小的一部分，更不是重要的。

具体的影响包括产品管理工作的开展，产品管理和其他部门工作关系的定义，个人产品管理工作技能的学习以及个人职业发展的规划和远景等。

平台上的差距：就个人而言，我认为这才是最为关键的。认知上的改变和工作上的差距，或许不是一时半会儿能改变的，但是建立一个能够正确引导、规范、职业化的统一交流平台则是可以在短时间内实现的。

一旦有了这个平台，那么就会对改变认知、提升工作起到很大的促进作用，而企业和

个人的变化则反过来又会促进平台的发展。

为什么我这里没有谈那些具体的工作技能和工作能力上的差距呢？因为我认为从一开始，我们就偏离了方向，再谈那些舍本逐末的工作毫无意义。

中国产品经理如何突破

如何才能缩小差距，赶上国外同行呢？

思路很简单，知道了问题所在，然后针对这些问题设计最适合的解决方案就可以了。

我的思路是，基于差距倒着做可能效果会更好。也就是说，要缩小差距，首先要做的就是建立一个提供真正产品管理服务的专业平台。

而这个平台的首要目的不是仅仅去提供一些信息、搞一些活动，而是要建立符合中国企业的产品管理体系、流程和方法。

然后基于这平台，再通过各种各样的形式（例如线上线下的交流、图书、教育等）把这些内容传递给企业和个人，一点一点地影响他们的认知和工作，最后形成统一。

其实不光是在IT行业，在国内很多行业都存在这样的问题，只不过是IT行业发展得比较迅猛，其表现也更为突出罢了。

但是，我们不要为此而感到灰心，虽然国外同行在认知、工作和平台上比我们要领先很多，但他们也不是每个行业、每个企业都发展得很好，同样也有不少企业面临着和我们一样的问题，如表3所示。

表3　国外同行面临的问题

原文反馈	中文翻译
We lost P&L responsibility at the product level and are well on the way to being a sales support function.	我们在产品层面上失去了管理产品生命周期的责任，几乎将要成为一个销售支持的角色了。
My original manager (VP) retired making a peer of mine a manager... big mistake!	我们那个早该退休的头（VP）使产品经理成为了一个排雷的，天大的错误！
I am spending far less time on sales calls (which is good) but also far less time out of the office (which is not good). We are “protecting” our product managers from giving sales demos but also making them spend far too much time on internal project meetings.	我花很少的时间在销售拜访上，而且也几乎不出办公室的门。他们说这是为了保护产品经理不被销售所占用，但是却花了更多的时间在内部项目会议上。
Less focus on “strategic”, more on “tactical”.	不注重战略，而只注重战术。
Hecticis the new normal—role has expanded beyond product management with no end in sight.	狂热成为一种常态，角色被延伸，超越了产品经理能够看得见的地方。

但是，我要说的还是那点：虽然问题是一样的，但我们找到解决方案了吗？至少国外同行找到了。

中国的企业在管理上有着一种莫名的自卑和对国外管理思想莫名的盲目，自卑造成我们没有勇气去探索适合自己的管理思想，而盲目则让我们不去批判地看待国外的管理思想，只是一味地照搬和模仿。

但是，我们一定要知道，一种管理思想对于企业来说，永远只能扮演“发动机”而不是“救命稻草”的角色，我们不要把任何管理思想看成是企业的快速消费品，短期内不见效就扔掉，而一定要把它当成耐用品，要用耐心、信心去看待你选定的管理思想。

但是，要把它当成耐用品，必须有一个前提，就是你是否用本土化、职业化和体系化的角度去看待它。

这应该就是我们在许多管理思想——包括产品管理——上无法和国外同行缩小差距的最根本原因吧！

（本文图表来源于UCPM 中国产品经理联盟。）

作者简介 | 马博

UCPM（中国产品经理联盟）发起人之一。现主要从事产品管理的研究与实践。曾经涉足重工、广告、会展、互联网、教育等多行业领域。个人邮箱：mabo@chinapm.com.cn。

■ 责任编辑：董世晓（dongsx@csdn.net）

一分钟先生
Mr. One Minute

程序员面试真经

作为面试官，在面试程序员时，都会关注哪些问题？而程序员应该从哪些方面做好面试的准备？本期话题相信对面试官和程序员都会有所启示。

刘秋伟
深圳市万兴软件有限公司研发总监

与其他岗位相比，程序员相对来说会内向一些，思维也会更严谨、更有个性。企业招聘程序员就是要求他能和团队一起完成既定的开发任务，所以重点从技能水平、学习能力、团队合作及工作心态等几个方面考察程序员。

技术水平是对程序员最基本的要求，很多企业会通过笔试来辅助考察。技能的考察主要看面试者技能匹配度、对技术发展趋势的了解以及自己的职业规划。

1. 你认为自己最擅长的技术是什么？

2. 谈谈你对XX行业/技术发展趋势的看法？对最近XX技术问题发表下你的看法？

3. 在五年的时间内，你有什么样的职业发展规划？

技术快速更替，员工的求知欲和学习能力比他现在的技能更为重要。宁愿招聘一个学习能力很强的员工，也不要一个靠吃老本、不愿学习的员工。

1. 你最近有看哪些书或参加过什么样的培训？有灌技术论坛的习惯吗？

2. 今年有什么学习计划？今年有什么目标？

3. 研究过开源项目吗？有什么收获？

在现在的软件开发中，已经没有所谓的“孤胆英雄”，项目主要靠团队合作来完成，而团队合作能力可以通过考察以往项目，来了解面试者对团队的理解、遇到问题的解决思路等。

1. 你做过哪些项目？最成功的是哪个？为什么？

2. 辛辛苦苦工作半年的项目失败了，你怎么办？

3. 评价下你过去的团队？你喜欢在什么样的团队中工作？

4. 你的业余爱好是什么？

招聘新员工，需要了解面试者对工作的心态和价值取向，并且与他沟通公司的企业文化。如果你所在企业需要快速发展，有高强度的研发任务，而他想找一份轻松的工作，可能就不是很合适了。

1. 为什么选择离开上一家公司？

2. 有了解过我们公司的产品/服务吗？

3. 你对加班的看法？你家人或朋友抱怨你加班怎么办？

4. 如果在试用期发现你技能不符合要求，该怎么办？

最后，技术人员的有效沟通能力也非常关键，特别是对问题的分析和说明，所以在整个面试过程中你要观察面试者的分析思路，对问题重点的把握以及表达。经常有一些滔滔不绝、但不知所云的面试者，让人非常头疼！

关注IT人自我管理、职业发展，广邀经验人士排疑解惑。

花一分钟阅读，还你一个明白

蒋建华

北方跃龙项目经理，微软最有价值专家（C# MVP）

程序员的能力分为两种：技术能力和非技术能力。技术能力包括编码能力、系统分析与设计能力；非技术能力包括写作能力、沟通与协作能力、组织与管理能力等。而根据程序员的能力水平，可以将程序员分成初级、中级、高级三个级别。因此，在面试过程中我会针对初级、中级、高级程序员三个级别来提出不同的问题。

在技术能力方面，重点考查初级程序员的编程能力、中级程序员编程能力并兼顾系统分析的能力、高级程序员的系统分析与设计能力。在非技术能力方面，重点考查初级程序员的沟通与协作能力、中级程序员的写作能力（主要指编写技术文档，如需求分析文档、用户手册、部署手册等）、高级程序员的组织与管理能力（如指导、协助中级程序员进行问题分析和开发）。

在面试时，程序员首先要将自己的技术能力展现出来，以一种交流的心态去面对，不要紧张，要有自信，即使面试失败自己也要下去总结失败的原因，找到技术弱点加以弥补；其次，面试中遇到不会的问题要讲明自己的思路，因为有些问题不是考你的编程能力，而是逻辑思维的能力；最后，对照程序员能力模型进行自我分析与评价，做好职业规划，不断学习，提高自己的编程能力和抽象思维能力。

以SQL Server为例，我来分享一下面试时常问的一些技术问题。

1. 聚集索引和非聚集索引有何区别？应该怎样使用？

2. SQL的一张表中有一个自增的ID字段，但是现在不连续，写一条SQL语句取出某个位置到另一位置（如50~60）之间的数据。

3. 如何判断和防止SQL注入？

4. 如何对海量数据进行处理？

第1题考查基础知识；第2题考查SQL语句的编写能力和技巧，属于初级程序员的问题；第3题考查SQL Server的安全性，属于中级程序员的问题；第4题考查SQL Server的性能，属于高级程序员的问题。通过这几个问题可以判断出应聘程序员的数据库水平。

毛颖

法国CAPS公司销售工程师

从个人经历来说，我会对以下五个问题比较感兴趣。

1. 请说出3个你觉得至今都没有算法可以解决的问题。

这是一个开放式问题。从回答中我们可以了解到被面试者的思维方式和思维敏捷度，而这两点是一名优秀程序员不可或缺的。我预料一般有两类回答：第一，学术类，比如有人会说一些关于寻找质数之类的现在还无法解决的问题；第二，生活类，我比较期待这一类有意思的回答，如果被面试者可以结合一些现在的社会现象作出回答，也可以从另一个角度反射出其社会属性。

2. 请从技术角度谈一下《黑客帝国》的观后感。

同样又是一个开放式问题，完全没有所谓的标准答案。《黑客帝国》是程序员的必看“教材”，整个故事最精彩的莫过于构思。我觉得能具备这样的构思能力是程序员的最高境界。那么从技术角度，对于这个构思的框架搭建以及实现等，我想听听程序员的不同理解和实现方法，从侧面了解他们的潜在创造能力和对身边事物的建模能力。

3. 你比较偏向于把程序员比作翻译、建筑师还是设计师？

问这个问题的目的是看你怎样理解程序员这个职业。一个人对自己职业的认识以及定位将

一分钟先生
Mr. One Minute

直接决定其在这个职位上的积极性和创造能力。我的答案是程序员既是翻译，又是建筑师，更应该是设计师。并且随着这三个职业的排序，正是我认为的程序员职业能力升华台阶。

4. 请设计至少两个不同的算法解决将一个蛋糕平均分为6份的问题。

比较实际的专业能力题。这是一个既简单又综合的考验。除了对程序员这个职位的正确认识之外，实际操作能力毕竟是影响今后实际工作的第一要素。这道操作题，考察应聘者对问题的思考，以及分析能力和解决问题的动手能力。

5. 比较一下这两个算法，你觉得哪个更好，依据是什么？

比较算法，还是考察的专业能力，目的是看你怎样判断一个算法的好坏。对这道题，没有接受过专业训练的应聘者应该无法给出全面且科学的分析。并且请他们评论自己设计的算法，也是对自我认识的一种衡量。

李颜杉

某外资人力资源顾问公司猎头顾问

程序员的概念有点宽泛，为更有针对性，我主要分享对“网络程序员”的观点。简言之，我们主要关注面试者的**硬条件**、**软条件**两个方面。

硬条件是指学历、技术经验、语言等能直接呈现的能力。技术是实在的东西，有就有，没有就没有，由不得半点浮夸。

我们比较感兴趣的问题如下。

1. 做过什么：是写代码、设计还是架构？
2. 做成功过什么：是大型的平台吗？其特质如可扩展、高并发、交互式。
3. 擅长做什么：使用的编程语言是哪种？使用经验有几年？
4. Debug的经验，遭遇什么复杂的问题（需例证其复杂性）？

技术面试的后续往往是笔试或上机，比如，在所有你使用过的Design Pattern中，请解释对比其中两种，如Command Pattern和Visitor Pattern（可以文字回答，也可以画图）？DOM parser与SAX parser的区别在哪儿？什么情况下采用DOM parser而不是SAX parser？大家在面试之前可以Google一下类似的题目。

如果面试外资公司，需要英文达到流利读写的水平。因为项目的完成往往需要协同全球多个国家的同事，如果缺乏英文技能，在外企的发展是比较受限的。计划今后进入外资公司的朋友，一定要加强练习。

软条件是指性格、兴趣、职业规划等非直接呈现的能力。公司对于程序员一般会看重两种性格特质：爱技术、乐分享。

“爱技术”的人才会自发的钻研，不浮躁，也才会在这一行做得长久。对于这点，面试官可能希望了解你工作之外喜欢做什么。我听到过很多分享：经营自己的博客或者技术论坛；尝试新技术、新设备，学习新技术是一种乐趣而不是负担；当然，下次你可以说你平时喜欢阅读《程序员》。

“乐分享”的人才能带动团队整体进步。对于这点，面试官可能希望了解你在团队中是什么角色。有的人可能讲我是Mentor，我们遇到技术问题时要内部讨论，我往往是能给出solution的人，所有人都解决不了，我们会求助于互联网——说明你不只愿意分享，而且有东西可以分享。

以前曾在面试中听说，程序员是青春饭，3~5年之后一定要转做Manager。我觉得职业发展一定要切合自己的性格，并不是每个人都很享受并善于处理人事管理。有的资深开发人员转成Manager几年之后，又转回纯技术（Individual Contributor）。而资深技术人员对于公司是非常宝贵的财富，不管从受重视程度、薪资福利待遇都是非常有吸引力的。

程序员成才的关键
——内在兴趣和善于发现

■ 文 / Peter Seibel　译 / 叶淮光

本文是Common Lisp专家Peter Seibel对计算机科学家Guy Steele的访谈，谈到了他程序人生开启的历程以及程序员成才的关键。

初涉编程

Seibel：你是怎样接触编程的?

Steele：嗯，当我还是个小学生时，我就已经深深迷恋科学和数学了，我读了很多这方面的书，比如Irving Adler的*Magic House of Numbers*，它是我的最爱。我也喜欢儿童科幻小说，比如*Danng Dunn*系列等等。总的来说，我对科学和数学有着广泛的兴趣。所有我能找到的关于科学和数学的东西，我都读了，同时我也读到了一点关于即将到来的新奇的计算机的介绍。

Seibel：你编写的第一段有趣的程序是什么?

Steele：嗯，我首先学习了Fortran语言，不过在我开始学习IBM 1130汇编语言之后，事情才变得真正有趣。我能想起来的最早的有趣的程序是一段能产生上下文关键字索引的东西。IBM为他们的用户手册提供一个被称作是快速索引的东西：给定一个关键字，你可以从一个按字母排序的索引中查找，关键字的前后是这个关键字的上下文的一些单词。

Seibel：你在MIT很自在，但最终却还是去了哈佛读书，而同时又在MIT打工，这是怎么回事呢?

Steele：我申请大学的时候，申请了三所学校，MIT、哈佛还有普林斯顿。我最想去的是MIT。三所学校同时都录取了我。波士顿拉丁学校的校长Wilfred L. O'Leary是个老派的学者。老先生人非常好，打电话给我父母说："你们知道令郎拿着哈佛的通知书实际上却考虑去MIT吗？"他就这样向我父母施压，我父母转而对我施压，最终我决定去哈佛了。

我父母继续找我的麻烦，让我去打一份夏季工，而不是在家待着——你知道，做父母的都会这样。我很清楚自己的兴趣是计算机，我可不想去快餐店摆弄汉堡包。我面试了打孔工的工作，并且自以为是完全能够胜任的。但是没有人愿意雇用我，部分原因是我还不满18岁，可找到后才明白。他们听了我的叙述后说："不要打电话给我们，我们会打给你的。"然后就杳无音信了。

大约7月初我听说MIT的Bill Martin正在寻找Lisp程序员。我想："啊哈，机会来了，我了解Lisp啊。"我过去经常出没于MIT的时候，从AI实验室搞到了一些Lisp文档的副本，我也曾偷偷溜进实验室摆弄过计算机。那些日子里实验室大门是敞开的，反越战抗议发生后门才被锁上。我在高中四年级时在IBM 1130计算机上实现过我自己的Lisp程序。

于是，我这个不知道哪里冒出来的小瘦猴儿，跑到Bill Martin的办公室，从门口探进头说："我听说你在招Lisp程序员。"他并没有嘲笑我，只是打量了我一下，然后说："你得先做做我出的Lisp考题。""没问题，现在考怎么样？"我就坐了下来，花了两个小时来答题。完成后我把试卷递给他，他用了十分钟浏览了一遍，然后对我说："你被录取了。"

编程导师

Seibel：在你起步时有没有遇到对你很重要的导师呢?

Steele：在拉丁学校期间我的数学老师

Guy Steele
著名计算机科学家，美国艺术与科学院院士，美国国家工程院院士，哈佛大学文学学士，MIT硕士和博士。1988年获得ACM Grace Murray Hopper奖，2005年获得Dr. Dobb's程序设计杰出奖，Jargon File的最初编撰者之一，《黑客词典》一书的编者。曾参与现存两种主要的通用Lisp方言——Common Lisp和Scheme的创建，目前致力于高性能科学计算新语言Fortress的设计。

对我的适当鼓励刺激很重要。9年级的Ralph Wellings，就是在那个感恩节周末借我书的那位老师，和我做了一个交易。他说："我注意到你在所有数学测验中都得到了100分。我可以让你在每周的前4天数学课都待在计算机室，不过在第5天数学课的测试上你必须得到100分，否则，交易就自动终止。"看，这就是激励。在那年余下的时间中我变成了考试高手——我特别刻苦地学习数学，因为这能让我接触到计算机。更好的是，第二年我的数学老师没有与我做同样的交易，这正好，因为我对那一年的数学了解不多。他们做出了恰当的评估。我的老师都是非常好的老师，我要学什么他们总是为我大行方便。

Seibel：在那之后，随着你更深入地学习计算机，有没有特别的人在这领域帮助你呢？

Steele：有，当然就是雇用我的Bill Martin。还有Joel Moses，他领导着Macsyma项目，我受雇于MIT期间就在这个项目组里。

Seibel：在整个大学期间，你一直在做这个项目吗？

Steele：是的，我在哈佛读书的时候就一直是MIT的一名雇员。在暑假时是一份全职工作，开学后它就变成了一份下午的兼职工作。我尽可能地把哈佛的课程安排到早上，这样我就可以搭乘地铁去MIT，用两三个小时来编程，然后再回去。

Seibel：一直用Lisp做Macsyma项目吗？

Steele：是的。具体说就是当Maclisp解释器的维护人员。Jon L. White原本同时负责解释器和编译器的工作。他后来成为了一位相当厉害的编译器大师，而我则负责解释器，这个分工不错。就这样，Jon L. White成了我的导师。Macsyma项目组里所有的人都很关照我。我也得以结识一些AI实验室的人，所以当我申请读MIT的研究生的时候，很容易就被录取了，因为他们已经了解我，并且知道我在干什么。

Seibel：你得到了计算机科学学士学位？

Steele：是的，我本来打算主修纯数学，并且都安排好了我的课程。后来发现我对什么无穷维巴拿赫空间完全没有感觉，简直要害死我了。幸好，我已经在业余时间学习了足够多的计算机课程，这让我可以在专业的时候很主动。确切地说，我转去修应用数学专业，而计算机科学是应用数学的一个分支，在哈佛应用数学又属于工程学的一部分。

Seibel：如果有可能让你重新学习编程，你会有什么不同吗？有什么事情你希望更早一点完成的吗？

Steele：并不是一开始在我的脑海里就有特定的目标。我对我选择的这条路也不后悔。回首往事，我想我是一个幸运儿，受惠于一系列有趣的巧合，或者说，恩赐。

现在我意识到，实际上同时在MIT和哈佛的经历是很不寻常的。我可以跑来跑去，然后说："这条河（编者注：哈佛大学与MIT只有一河之隔，即查尔斯河）那一边的教授是这样说的。"而这一边的教授就会说："哦，别信他，你应该这么想。"这很快让我的视野更开阔。

作为高中生就能进入MIT是另一个相当不寻常的经历。我15岁时就可以摆弄那些价值数百万美元的机器，在那时1百万元可真是一笔相当大的钱。所以，我当然没有抱怨，没有后悔，也不会有任何得陇望蜀的想法。我本性也是个随遇而安的，既来之，则安之。

Seibel：与那时相比，对于编程的思维方式，有哪些大的改变？除了认识到冒泡排序不是最好的排序算法之外。

Steele：我想对我来说，最大的变化是认识到你不可能了解运行在你计算机上的所有一切。有些事情绝对超出了你的控制，因为不可能了解所有软件的一切细节。而在上世纪70年代计算机仅仅有4K字节的内存，你完全可以做一个内存转储，然后一个字一个字地去检查是不是你期望的。阅读操作系统的源码，了解它是如何运行的自然也正常。我也确实这样干过——我研究过磁盘管理程序和卡片阅读机程序，然后实现了我自己的版本。我觉得我自己了解整个IBM 1130是如何运作的，或者至少可以说我了解我自己想了解的所有事情。不过现在你再也不能这样干了。

编程参考书

Seibel：在你学习编程的时候，有没有什么书对你特别重要？

Steele：在70年代的书当然是Knuth的TAOCP《计算机程序设计艺术》。

Seibel：你从头到尾地读过吗？

Steele：差不多每页都读过，差不多。我做了几乎所有我能做的习题。一些被称作是高等数学之类的东西我不太明白，我做了些注释或跳过了那些我不懂的。不过头两卷和第三卷的大部分我都很认真地读过。还有Aho、Hopcroft和Ullman编著的那本算法书（编者注：是指*The Desing and Analysis of Computer Alogorithms*（《算法分析与设计》）一书）——我想我是从这里面真正学到如何排序的。我得从我的书库里查找一下，看看能不能还记得有什么别的书。我是个收集狂——这类书我都有。不过这两本是我首先想到的。还有Lisp的书。Berkeley和Bobrow编辑的III Lisp（编者注：III指Intformation International，Inc.）：是主题各异的论文集，不过我从中学到了很多有意思的东西。然后我开始读《SIGPLAN公报》和《ACM通讯》。那些日子里的《ACM通讯》可是有很多真正的技术内容，非常值得一读。

我要提及两件事。第一件，当我在拉丁学校开始对科学感兴趣的时候，我决定从事计算机科学相关的事情。有一天有个导师问我："你考虑过成为ACM的学生会员吗？"我不知道他的名字。不过我从那时起就非常感谢他，这给了我很大的鼓励。

而我上哈佛以后，每当早晨有点空闲时间，我就会去Lamont图书馆做两件事：按照我的方式，从后往前阅读《科学美国人》，或者从前往后阅读《ACM通讯》。对《科学美国人》，我特别注意Martin Gardner的所有数学游戏专栏。对《ACM通讯》则阅读所有我感兴趣的文章。在1972年这本期刊还只有15年的历史，所以不难把它们全部都过一遍。

Seibel：阅读所有文章在那时比在今天要容易得多，一个人还是希望了解整个领域的。

Steele：是的，你有希望了解这个领域。有很多只有一页长短的文章，让你知道："这儿有一项新的散打技术。"我读了很多这类的文章。

Seibel：旧的文章我个人常常觉得不太容易理解，因为它们和一些旧硬件或者旧语言联系得比较紧密。

Steele：是这样的，需要是创新之母——一个想法的出现是因为在一个特定的环境下需要它。过了一段时间，大众认识到这个想法很重要。然后你需要摆脱环境的局限，展现出核心思想本身凸显出来，这样就可以流行好几年了。"这个神奇的技巧可以按位逆转字。"他们给出了7090汇编语言的一些东西。有些很有趣的数学思想在里头，不过他们还不能从中抽象出来。

Seibel：的确很多人从学校里开始，在指导下学习计算机科学知识。但是还有很多程序员是没有正规学历背景的，只是边干边学。对这个你有什么建议吗？你怎么开始阅读那些技术论文，如何抓住要点并理解它呢？应该从ACM最初读起，一直读到现在吗？

Steele：嗯，首先，我得说通读《ACM通讯》并不是我刻意博览群书成为一名伟大的计算机科学家的计划。我阅读是因为我有兴趣，有内在的动力去学习那些资料。所以我觉得这里有两个因素：第一是有内在的动力，想要阅读它们，因为你有兴趣或者说你觉得能提高你的技能。

通读《ACM通讯》并不是我刻意博览群书成为一名伟大的计算机科学家的计划。我阅读是因为我有兴趣，有内在的动力去学习那些资料。

另一个问题是你怎样才能发现好的东西？当然，对"好"的认识也是三十年河东，三十年河西。今年你觉得是真正好的十年后说不定过时了。我觉得你可以拜访一位曾经经验丰富的前辈，问他觉得什么才是好的东西。对我而言就是Knuth，就是Aho、Hopcroft和Ullman。还有Gerald Weinberg《程序设计心理学》，那本书今天还是非常值得读一读的。Fred Brook的《人月神话》也给我一些启示。

那时候我流连于MIT书店的计算机科学书架，下决心每个月到那里去一次，去翻翻那些书架。今天你再去一个书店，它的计算机书架规模可能是那时的10倍了，不过其中大部分图书都是如何使用C或者Java的。但还是会有一部分理论背景、算法这类书。

代码阅读

Seibel：另一种阅读——我知道你认为很重要的——是代码阅读。你是怎么样以你的方

式切入不是你编写的一大段代码的呢?

Steele：如果那个软件我知道如何使用，但不了解内部的工作机制，我通常会选择一个特定的命令或者交互行为然后追踪下去。

Seibel：执行路径吗?

Steele：是的。如果我要开始阅读Emacs源代码的话，我会说："让我们看看'向前移动一个字符'的那部分代码吧。"即使我不能完全理解，我至少会知道它使用的一些数据结构以及缓冲区是怎么表示的。如果我足够幸运，我能找到缓冲区增加一个的地方。一旦我理解之后，我接下来会尝试"后退一个字符"、"删除一行"。通过我的方式就了解越来越多的使用方法或者交互，直到我觉得我能够按照这种方式追踪代码的其他更重要的部分。

Seibel："追踪"是指查看源代码在心里执行它呢，还是要在调试器中启动它，然后单步执行进去呢?

Steele：两种方式我都会做，我会用单步调试器对付那些70年代或者80年代的小一点的程序。今天的问题是从启动程序到它真正可以做点什么，这中间有一段很长的初始化过程。所以更好的办法是找到主命令循环或者中央控制子程序，从那里开始追踪。

Seibel：当你找到了那些以后，你是设置一个断点然后单步跟进去呢，还是仅仅在脑海中想象它们的执行过程?

Steele：我更愿意做桌面检查——就是阅读代码想象它会做什么。如果我确实需要理解整段代码，我会坐下来试图按照我的方式来通读代码。不过你不能一上来就这样做，应该先在脑子中有了事情的组织框架。现在，如果你足够幸运，程序员会留下一些文档或者规则的命名，或者合理组织文件的顺序，方便你快速阅读他们的代码。

合理组织文件的顺序

Seibel：什么是合理组织文件的顺序?

Steele：很好的问题。使我想起了诸如Pascal这样的程序语言的一个问题，Pascal是为只过一遍的编译器设计的，源文件中的过程倾向按照自底向上的方式组织，因为在使用过程之前你必须定义过它们。也就是说，阅读Pascal程序最好的方法实际上是从后面读起，因为这样你就会看到程序自顶向下的结构。现在（编译器）形式如此多样，你也就不能指望什么了，除非程序员有一颗很强的责任心，将一切事情安排得井井有条，有助理解。不过，第三点，我们现在也有很棒的IDE来帮助你查看交叉引用，也许程序的线性组织也不再是那么重要了。

第四点，我个人非常不喜欢IDE的一个原因是，你看完了所有内容后，还是很难理解。在图形迷宫里乱窜，很难知道所有地方都走到了。但如果你得到的是线性顺序，就会确保所有事情都会梳理到。

Seibel：那么在你写代码的这些日子里，你是不是尽量按照自顶向下来组织代码呢?高层函数出现在它们依赖的低层函数之前?

Steele：我尽量表现高层的想法。最好的表现方法可能是展示一个中心的命令，控制的过程，以及向下面分发的事情。或者，重要的事情可能是首先要展现数据结构，或者说较重要的数据结构。重点是要像讲故事那样去表现想法，而不是只罗列一堆代码在那里。

在MIT工作时，一件很棒的事情是可以随意访问到没有加密的代码，它们全是非常聪明的黑客作品。于是我读过了ITS操作系统，也读过了TECO的实现和Lisp的实现。还有第一个相当漂亮的Lisp格式打印程序，是Bill Gosper编写的。事实上，我在读高中的时候就读过它们，还试图复制一些到我个人的1130实现当中。

如果没有接触到在其他机型上的现有Lisp实现，我一定不能为1130实现Lisp的，我根本就不知道怎么做。这是我个人教育的一个很重要的部分。时至今日，我们面对的部分问题是软件已经变得很有价值了，大部分软件都是商用的，也就是说我们不再有可作为免费例子的优秀代码来参考了。开源运动在某种程度上扭转了这一点。如果你愿意，你可以深入阅读Linux的代码。在我那个时候阅读TeX的代码是一项很有意义的练习，因为它是一大块良好组织的、易于调试的代码。

本文节选自人民邮电出版社北京图灵文化发展有限公司出版的《编程人生》一书。该书是当今15位大师级计算机程序员的访谈录，重点介绍了他们的编程感悟。特此感谢图灵公司授权。

面向接口编程的魅力

■ 文 / 杜玄　苏丽辉

面向接口编程已经是面向对象设计中人们的共识，而作者对面向接口编程魅力的体会，还要从一个旧软件项目的改造谈起……

最近，我接到一个关于旧的报表软件系统改进的项目。这是个用Java语言实现的单用户图形界面报表软件系统，已经运行了将近10年。Java在当时是一种比较新潮的语言，用Java完成报表系统的开发还不是特别多见。整个软件系统采用经典的两层结构框架，也就是说，在图形用户界面代码中直接调用JDBC连接数据库。体系结构如图1所示。

图1　两层结构的报表软件系统

问题的提出

随着业务的发展，用户在原有系统的基础上不断提出新的需求。一方面，由于积累了用户提出的各种各样的数据报表需求，旧报表系统已经支持了大量的数据报表；另一方面，用户又提出了其他需求，要求报表系统可以同时支持多个用户，且多个用户可以远距离访问报表系统。面对这些新需求，旧报表软件系统的体系架构显然已经不能满足要求。旧报表软件系统必须进行体系框架的改造，才能满足用户的新需求。从技术角度简单来说，新需求的任务就是要把两层结构的报表系统改造为具有报表服务器的多层体系机构，并且能够将这个报表软件系统部署到互联网上。系统改造的目标系统架构如图2所示。

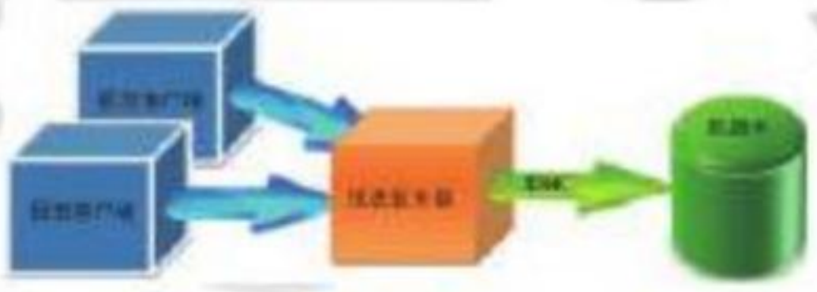


图2　多层多客户端的报表软件系统

多层体系结构在现代软件系统中已经得到了广泛的应用，其好处这里就不再赘述。建立多层体系架构，我们完全可以借鉴现成的框架完成这项工作。技术上，这并不是问题。经过近十年的积累，报表软件中已经支持了大量的数据报表，并且旧系统中报表数据处理逻辑与界面逻辑完全混在了一起。我们不可能把旧系统完全推倒而建立一套全新的报表软件系统。一方面时间不允许，另一方面大量的人力和物力投入也是无法实现的。事实上，如何把业务逻辑从旧的报表程序中剥离、分离客户端和服务端，才是最大的难题。这个任务的关键在于，如何以最少的代价完成旧系统到多层体系框架的迁移，而不需要处理业务剥离的任务。

抛开迁移所需要的工作量问题，我们只从体系框架的升级来分析这个项目。对于一个时间紧急的软件项目，引入第三方实现、在第三方软件的基础上完成旧系统的迁移应该是一个比较好的选择。这里，首先要注意的是要有合适的第三方软件实现。再者，要评估第三方软件学习曲

线是否太长，而导致新软件很难学习而耽误开发周期。同时，应当注意的是第三方实现对于商业用户而言通常会有诸多限制。尤其很多开源软件，并不适合商业用户的应用。比如当前的这个项目，我们必须要有源代码的控制权，必须能把第三方工具集成在我们的产品中再发布。在Java世界，开源的报表工具很多是基于GPL和LGPL协议的，比如iReport、jasperReport和JFreeChart等。它们是很好的实现，用起来也很顺手。但是对于正规的商业软件公司来说，开源并不等价于免费，尤其对商业应用来说更是如此。简单地说，开源软件如果基于GPL协议就是说“我开源，你用我，你就也要开源”，LGPL协议就是说“你可以用我的库，但是你不拥有代码控制权”。对于商业用户而言，Apache和BSD协议才是友好的协议，因为基于Apache和BSD协议的开源软件可以自由地在商业项目中应用和修改。然而，实际情况是，基于这两种协议的免费报表工具确实不多。

解决思路

开源软件的代码不能随便引用，我们就开始考虑自己动手搭建这个软件框架。由于旧软件系统是通过JDBC连接数据库的，所以我们就从这个标准化的JDBC为入口来考虑，作为解决问题的突破口。基于这个思路，我们可以利用编写代理JDBC驱动程序完成两层系统到三层系统的迁移。接口的魅力从此展开，方案的总体结构如图3所示。

图3　基于JDBC驱动替换的迁移框架

1997年，Sun公司在JDK1.1中引入了数据库接口API，那就是JDBC（Java Database Connectivity）。JDBC屏蔽了不同数据库的细节，为应用程序提供了统一的访问方法。JDBC就是访问数据库的规约，只要是符合规约的实现就可以被其他程序透明地调用。JDBC提出了四种数据库连接类型，其描述如下。

类型1：由Sun公司内置提供的JDBC-ODBC桥形式进行连接，是作为JDK1.1后的一部分，当然也是sun.jdbc.odbc包的一部分。

类型2：利用开发商提供的本地库直接与数据库通信。

类型3：纯Java实现的驱动，经过服务端的中间件层连接数据库。这种方式具有最大的灵活性 。

类型4：纯Java实现的驱动，直接用数据库内部协议连接数据库。这种方式性能较高。

根据JDBC类型的文字描述，结合旧软件系统的现有情况，我们可以确定上述的类型3就是我们要实现的体系结构。**现在问题归结为：编写自己的JDBC驱动程序和搭建自己的中间件软件层。**

图3描述了基于类型3的JDBC实现。在客户端，旧程序是调用Oracle的JDBC驱动访问数据库的。在这个系统结构中，把原有的Oracle的JDBC驱动替换为“数据库驱动代理”。这个“数据库驱动代理”模块是一个JDBC驱动实现，是遵照JDBC驱动接口实现约束来实现的。“数据库驱动代理”不直接访问数据库，而是通过“代理客户端”将数据库操作请求通过网络发送到服务器端的“代理服务器”模块，也就是服务端的中间件模块。“代理服务器”模块再调用真正的JDBC驱动（Oracle的JDBC驱动），完成对数据库的访问。数据库的返回结果再通过“代理服务器”返回给“代理客户端”，再由“数据库代理驱动”返回数据给报表应用程序界面。

这个新系统的时序过程是一个非常简单的直通过程，只是增加了几个中间层，中间的通信由本地调用转为跨越网络的调用。在此不再赘述。

基于图3所示的迁移框架，原有的报表程序只需要极少的代码修改，就可以完成代码的迁移；相应的，旧代码中只需要简单地替换JDBC驱动加载代码就可以完成两层结构到三层结构的转化。如图4中的代码示例，改写两行代码就可以完成旧程序的改造。红色方框中注释掉的是旧的Oracle JDBC驱动加载，替换为新的XduDriver驱动加载就完成了对旧代码的修改。很显然，这样的修改对旧系统业务逻辑一点也没有影响，简单而快速。

根据Sun的JDBC驱动开发文档（http://java.sun.com/products/jdbc/driverdevs.html），如果要

```
//Class.forName("oracle.jdbc.driver.OracleDriver").newInstance();
//Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:ORCL","SCOTT","TIGER");

Class.forName("XduDriver").newInstance();
Connection conn = DriverManager.getConnection("jdbc:xdu@localhost:1521:ORCL","SCOTT","TIGER");
```

图4 JDBC驱动的替换

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class ClientTest {

  public static void main(String[] args) {
    try{
      Class.forName("XduDriver").newInstance();
      Connection conn =
          DriverManager.getConnection("jdbc:xdu@localhost:1521:ORCL","SCOTT","TIGER");
      String query = "SELECT * FROM BONUS";
      Statement st = conn.createStatement();
      ResultSet rs = st.executeQuery(query);

      while (rs.next()) {
        String strVersion = rs.getString("INSTALLED VERSION");
        String strDate = rs.getDate("INSTALLDATE").toString();
        System.out.println(strVersion + ",  " + strDate);
      }
      conn.close();
    }
    catch(Exception ex){
      ex.printStackTrace();
    }
  }
}
```

图5 一个简单的JDBC查询

实现一个JDBC驱动，这个实现必须遵守JDBC驱动的接口。

JDBC驱动实现必须完全实现的接口如下：

```
java.sql.Driver
java.sql.DatabaseMetaData
java.sql.ResultSetMetaData
```

上面的接口必须实现，但有些方法可以选择实现，这依赖于数据库管理系统是否支持：

```
java.sql.CallableStatement
java.sql.Connection
java.sql.PreparedStatement
java.sql.ResultSet
java.sql.Statement
```

JDBC驱动需要实现的接口并不复杂，而且报表系统只需要查询数据库系统，而不需要修改数据库系统中的数据，所以这里可以最小化地实现JDBC驱动程序代理。旧报表程序只用到了很简单的JDBC查询功能，也就是用Select这样的SQL语句查询数据库。其典型的代码示例如图5所示。

虽然JDBC驱动开发文档要求必须实现某些接口和方法，并且有些接口中的方法很多，全部实现起来工作量也不小。但是，我们可以根据旧报表系统的情况最小化地实现必要的JDBC驱动接口。如图5的JDBC调用代码所示，报表软件系统只需要实现画红线的方法就可以完成这个查询过程。基于这个思路的JDBC驱动实现参见图6中的类图。

图6中，蓝色类XduDriver、XduConnection、XduStatement和XduResultSet所提及的方法必须实现。根据JDBC驱动实现要求，在XduDriver中还要实现一个静态块，其功能就是注册XduDriver这个JDBC驱动到Java的DriverManger中。这里只需要一条语句java.sql.DriverManager.registerDriver(new XduDriver())。绿色类不需要编写代码，可以用Eclipse工具自动生成方法中的缺省实现，因为绿色类图中的方法在简单查询应用中没有被调用，所以只要能够编译通过就可以，不需要编写实现逻辑。

一个基于RMI的实现

因为RMI可以实现查询请求数据和查询结果数据在网络上的序列化承载，所以代理客户端与服务器之间采用RMI来实现通信。用RMI完成这样的工作是非常方便的。基于这样的思路，系统实现的类关系如图7所示。XduResultSet类承载查询结果数据，并且要通过

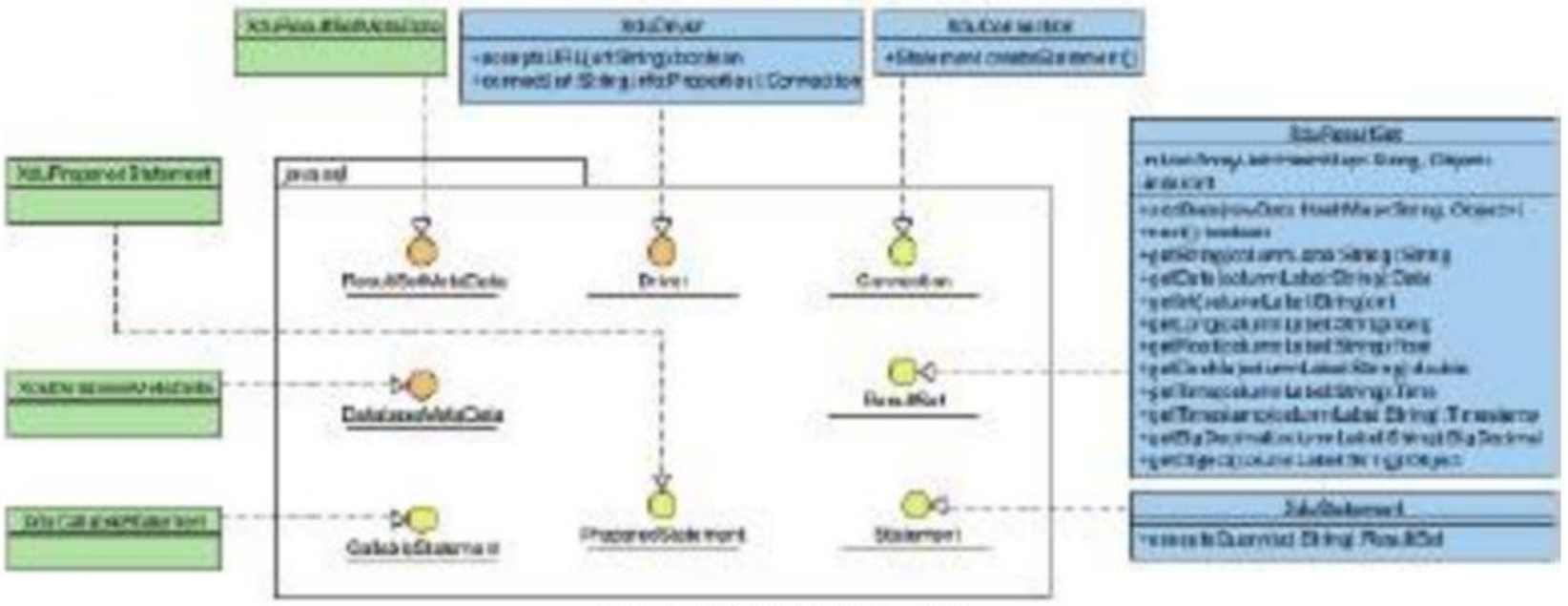


图6 JDBC驱动基本接口实现

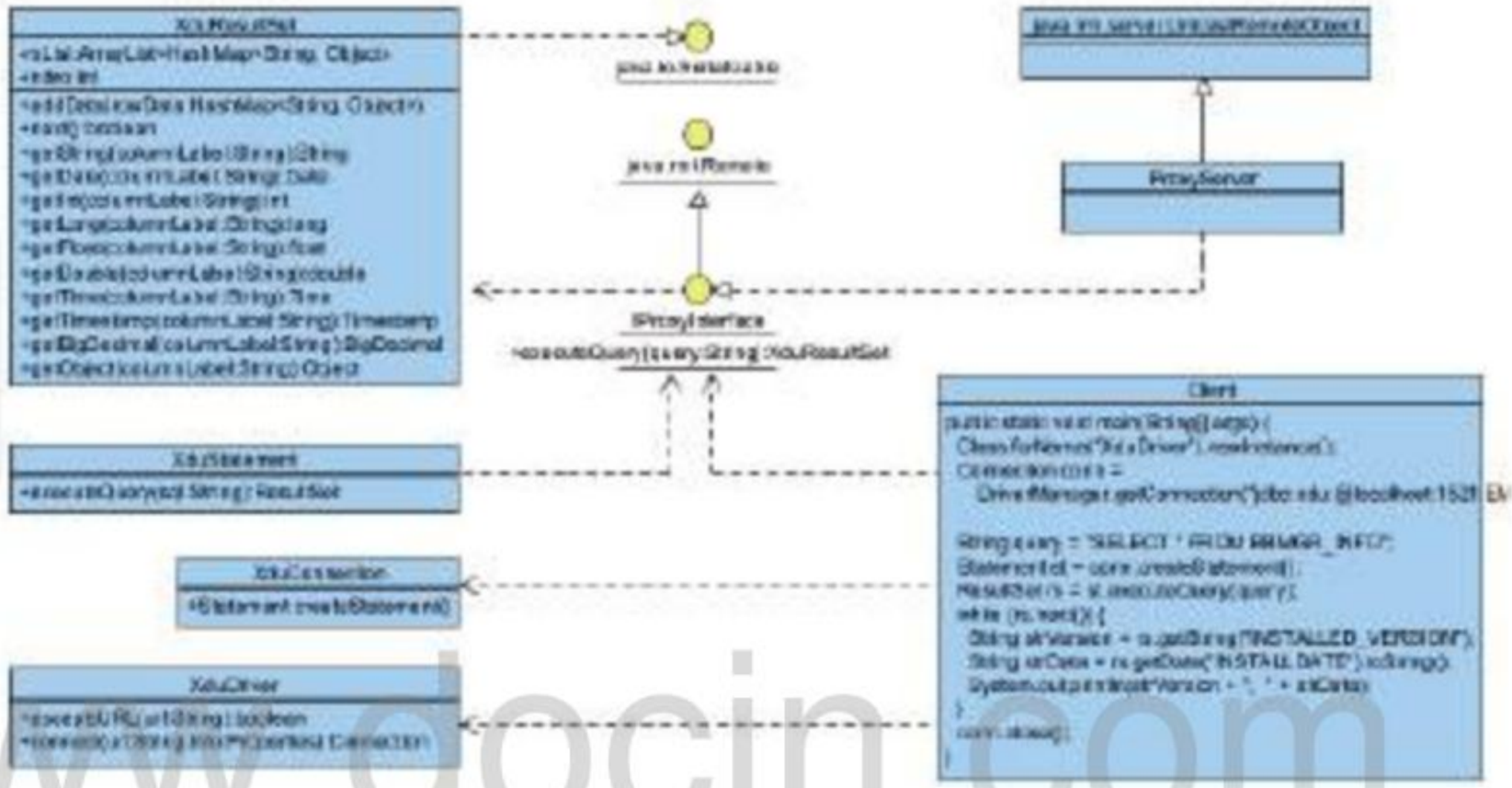


图7 基于RMI实现的类图

网络完成序列化传输，所以XduResultSet必须实现Serializable接口，以支持对象在网络上的序列化。IProxyInterface是RMI客户端和RMI服务器之间的接口，定义了executeQuery方法，负责查询请求和查询结果的交互。

这个实现示例的调用过程非常简单，图8中的时序图可以比较清楚地反映调用时序。

JDBC Over HTTP

上面的改造把旧报表系统改造为一个三层结构的报表系统，虽然还是一个胖客户端的软件系统，但确实完成了两层到三层结构的迁移。由于这个改造的系统是用RMI为通信载体，就使得该迁移方案无法完成在互联网上的部署。因为企业网与外网连接时，为了保证内网的安全，通常采用防火墙完成内网与外网的隔离。RMI如果要穿过防火墙，必须在防火墙中开放特定的端口。但是，一般防火墙只会开放HTTP协议的80端口，其他端口则会拒绝服务。这就意味着，旧报表系统虽然迁移到了三层结构，但并不能完成在互联网上的部署。在这里，我们展开思路，看看如何才可以使JDBC穿过互联网，使旧的两层系统可以实现更广泛的部署。透露一下，其实这里完全可以借用我那篇名为《突破防火墙——基于HTTP协议的Java隧道通讯》（注：原文链接请见http://blog.csdn.net/seave/archive/2006/04/02/648345.aspx）的文章提到的技术，来完成“JDBC over HTTP”。下面我们看看这个系统的基本构架，如图9所示。

Java隧道技术是用现有的Web Server和Servlet容器建立的Java消息隧道和远程方法调用的方法。由于是基于HTTP协议的，所以Java隧道是防火墙透明的。Java隧道技术可以把Java对象通过HTTP协议实现序列化，实现Java数据在

互联网两端的通信传递。

借用Java隧道技术，只要在原有的RMI体系结构上，增加Web Server以提供基于Servlet的Java对象隧道服务。在客户端将JDBC代理的查询请求对象包装为HTTP协议的数据，再传递到隧道服务端。隧道服务端将HTTP数据逆向变换为Java对象，再传递给代理服务器。数据库的查询结果数据以相反的过程完成工作。数据库驱动代理和Java隧道技术，使旧报表系统完成了互联网上两层到多层的迁移。

扩展开来，这种迁移方式不但可以用于Java语言基于JDBC接口的数据库程序，也可以用到ODBC等类似的旧数据库程序上。通过对数据库驱动程序的替换，以很小的代价完成旧数据库系统在多层次体系构架下的发布，当然，也可以更进一步地实现互联网上的部署。比如我们可以做到“ODBC over Web Service”，基本的原理与本文的思路并无不同，只不过所面对的是ODBC驱动和Web Service的实现罢了。

测试中的应用

JDBC驱动替换的方法，不但可以用在旧系统的改造工作中，而且也可以用在数据库相关软件模块的单元测试工作中。实现JDBC驱动代理作为测试桩，在测试桩中提供数据容器。用户在进行测试前，可以在容器中准备期望的数据，当数据库模块通过JDBC驱动代理查询数据时，测试桩返回事先准备好的数据。测试程序再根据返回数据与期望数据比较，从而完成模块的单元测试。

我们曾经提出了“一种数据库依赖软件单元的测试方法”，其基本原理是利用内存数据库（如HSQLDB）代替实体数据库（如Oracle、SQL Server等）。利用JDBC的接口一致性，用内存库的Connection替代实体数据库的Connection，完成数据库相关模块的单元测试。而这里，我们可以用一个新的思路来解决数据库模块脱离实体数据库的单元测试问题，那就是实现自己的JDBC驱动，作为测试桩，同样可以完成脱离实体数据库的单元测试，而且更进一步地脱离了对内存数据库的依赖。

综上所述，我们证明了“面向接口编程”带来的好处。由于旧系统是对JDBC接口的依

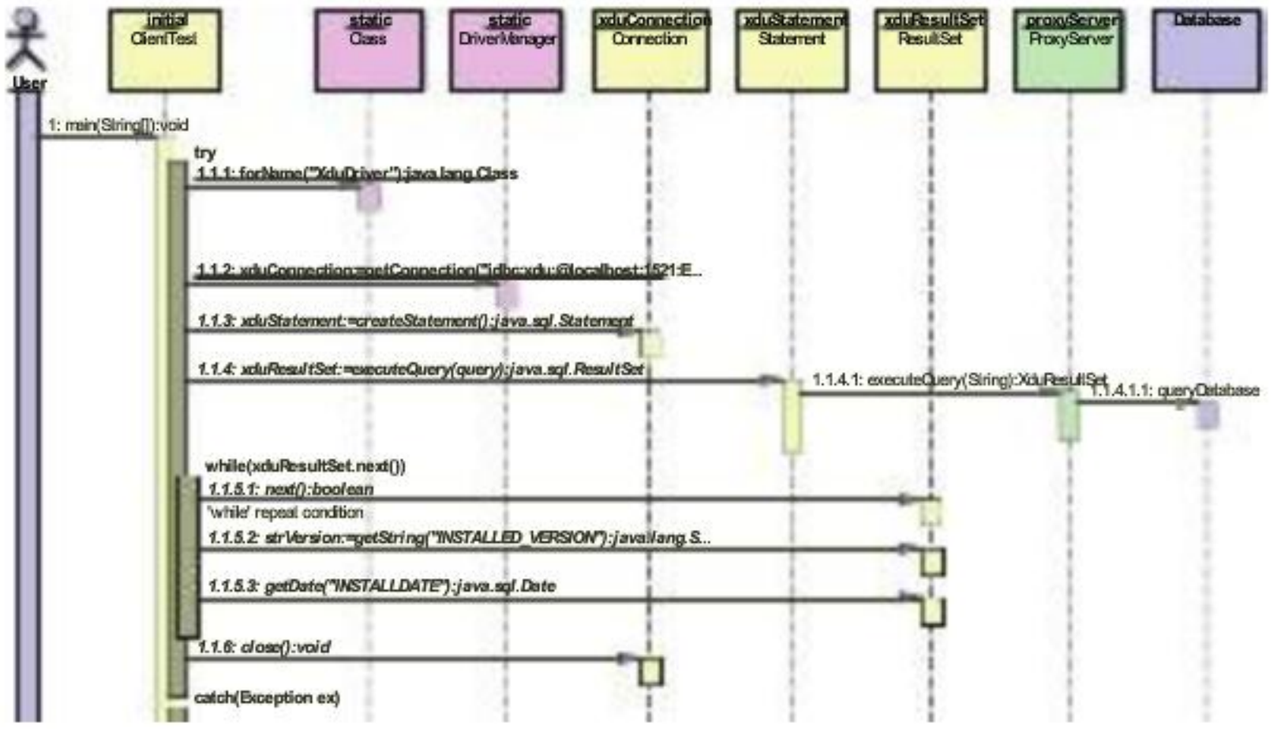


图8　实现示例调用关系时序图

图9　基于HTTP协议的系统迁移框架

赖，这样我们才有机会进行这样的改造迁移。通过接口将问题域抽象化和将细节隔离，把对具体对象的依赖转化为对接口的依赖。接口内的变化不会影响依赖接口的模块。JDBC作为一个规范的接口，带来的好处在这个旧项目的改造中完美地体现了出来。在此，我们体会到了“面向接口编程”的魅力。

作者简介 | 杜玄　苏丽辉

杜玄，Tellabs高级软件工程师。多年从事软件研发工作，对需求分析、软件设计、系统 建模、软件测试、项目管理均有涉猎。最近对软件研发心理学感兴趣。

苏丽辉，EMC（中国）上海研发中心高级软件工程师。有多年Java、C#编程经验，专注于提高软件质量，对软件设计、软件流程有浓厚兴趣。

■ 责任编辑：董世晓（dongsx@csdn.net）

Golang初探

■ 文 / 李兆海

本文以一个简单的例子，带领读者领略Google的新语言Go。主要探寻了Go语言对并发的支持以及独特的面向对象实现和基本语法。

Go语言是Google于2009年11月公布的一个新语言项目，其目标是创造一门既简单又有效率的开源编程语言。由于有C语言创始人Ken Thompson的参与，Go一面世，就被看成是C语言的继任者，受到很大关注。Go一方面吸收了C语言简单清晰、执行效率高的优点，另一方面融合了动态语言的闭包、动态绑定等特性，更加适应目前多核与多机高并发的开发环境和快速敏捷的开发效率。此外，Go并没有跟随主流的以“类和继承”为基础的面向对象实现方式，而是以接口和动态绑定的方式，将封装的粒度做得更细、更灵活，实现了另一种面向对象的代码组织形式。

本文将带领大家用Go来实现一个简单的程序。程序本身是对MapReduce的一个模拟，将一组数字交给一组并发的DoubleNode节点做翻倍，然后再由一个SumNode将翻倍后的数累加并输出。节点间的关系类似如图1所示。

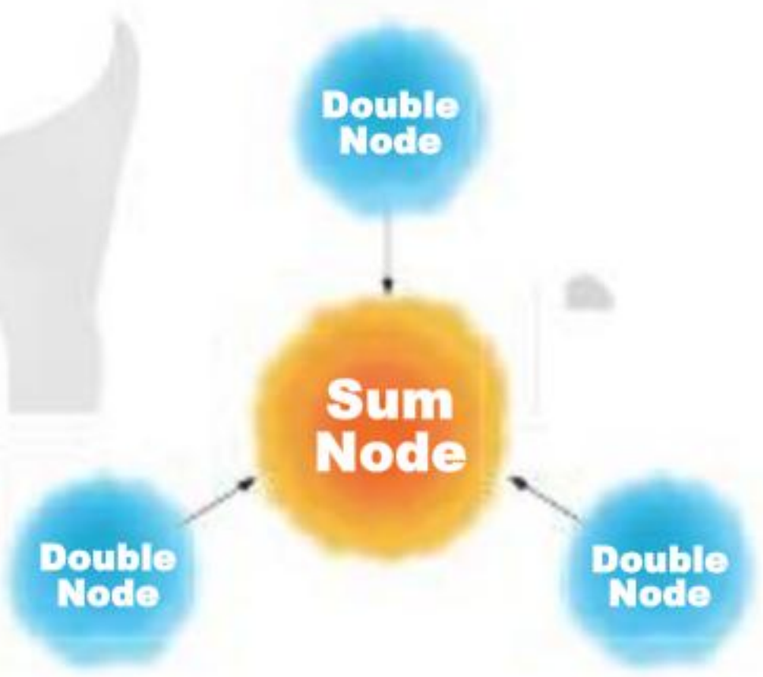


图1 节点结构图

Node的实现

图1中的每个Node都是一个独立运行的节点，节点间是并行的。值得提醒的是，Go提倡通过通信来共享数据，而不是通过共享数据来通信。因此每个节点既不访问别的节点的内部状态，也不访问全局变量，节点间通过Go语言的通道机制互相传递消息，通知工作完成并将数据传递给下一个工作节点。

确定Node的结构

首先是确定Node的结构。DoubleNode和SumNode都作为Node的一种特化，不同之处在于在Node执行时执行的功能不一样。统一起见，为Node定义如下接口：

```
type NodeInterface interface {
  receive(i int)
  run() int
}
```

蓝色的部分是Go的关键字，type表示定义一个新的类型，语法上与C语言的typedef类似，只是将被定义的类型名字和类型的顺序颠倒了一下。Go里所有涉及类型名/变量名和类型的地方，使用顺序都和C是颠倒的。NodeInterface是新类型的名字。interface表示定义的是一个接口，这个接口可以类比C++的纯虚类或者Java的接口，但是在使用的时候是动态绑定，不需要实现者必须继承自接口（后文有更详细的说明）。接口内定义了两个函数receive和run，其中receive接受一个参数i，类型是整数类型int，没有返回值，用处是处理从其他Node接收到的消息；run没有参数，返回值为int，用处是进行Node自身的运算。

定义Node内部的状态

另外定义Node内部的一些状态：

```
type Node struct {
  name string
  in_degree int
  in_ch chan int
  out_ch chan int
  inode NodeInterface
}
```

与前面不同，Node的类型是struct。和C语言的struct一样，Go的struct里只含有变量，不能有函数。Node类型里一共定义了以下变量：name，字符串类型，用来存储标示Node的名字；in_degree表示一个节点的入度，也就是本节点需要从多少个其他节点接收数据，整数类型；in_ch和out_ch是输入和输出管道，类型是传输整数的chan，chan的概念先按下不表，后面在使用的时候会讲到；最后是inode，类型是之前定义的NodeInterface接口，用来特化Node的行为。

这里值得注意的是，Node使用了类似模版模式的概念，但和C++/Java不同，并没有从NodeInterface继承，而是将NodeInterface作为一个成员。由于Go无法让一个类的成员函数处于未定义的状态，因此无法像C++/Java一样借由在子类特化父类里未定义的函数来实现模版模式。不过，这样虽然看似麻烦，但是好处在于将Node本身的状态和NodeInterface分离，两部分责任更清晰。

Node相关的方法

之后是Node相关的方法。首先是Node的创建方法：

```
func NewNode(name string, inode
  NodeInterface) *Node {
  return &Node{name, 0, make(chan int),
    make(chan int), inode}
}
```

func关键字表示接下来要定义一个函数。函数的名字是NewNode，接受两个参数：字符串name和NodeInterface接口inode。初始时，节点Node的入度in_degree为0。系统函数make会创建一个chan int的实例，并返回其引用。关于make的更详细的用法和限制，可以参考Go语言的官方网站。Node{...}一句表示创建了一个Node实例，花括号内按顺序给出Node结构内部变量的初值。然后用&运算符取新建实例的地址，作为返回值。是的，Go里依然有指针的概念，而且这个指针和C的指针概念类似，相关的语法也类似。

Node间需要互相连接，以下是连接的实现：

```
func (from *Node) ConnectTo(to *Node) {
  to.in_degree++
  go func() {
    i := <- from.out_ch
    to.in_ch <- i
  }()
}
```

函数名前的(from *Node)表示ConnectTo函数是类型Node的一个成员函数。与C++/Java不同，Go里没有类似this的关键字，在声明函数时需要明确指定指向当前实例的变量名。每个链接，都会增加被指向Node的入度数。go关键字启动一个gorountine，等待前一个Node的输出，并将输出的内容传入后续的Node。go关键字之后是一个匿名函数并执行这个函数。可以参考下面这种定义：

```
f := func () {...}
f()
```

如果将中间变量f去掉，就是上面提到的定义一个函数并执行的写法：

```
func () {...} ()
```

Go语言里的函数地位和变量是一样的，可以任意赋值给一个变量，有自己的生命周期，并且在其他函数间相互传递。而且Go的函数支持闭包，在一个定义域里定义的函数可以直接引用外层定义域的变量并在这个函数的生命周期里一直保存。不过要注意的是，如果闭包引用的是一个指针，需要小心操作这个变量，因为函数里和函数外的指针指向的是同一个地址，任何对这个指针指向的实例操作，都会对所有指针有影响。

关键字<-是对chan类型独有的操作。之前说过，chan类型类似于通道，可以把一个数据放进去，并在之后取出来。在例子里，<- from.out_ch是从from实例的out_ch通道里取出一个数，如果通道里没有数，则会阻塞等待。取出数后会把这个数赋值给i。之后将取出的值i通过to.in_ch <- i传入到to实例的in_ch通道里。这样就完成了将from和to两个节点连接起来的功能。

gorountine

gorountine是Go语言里很重要的新概念，有点类似线程，但消耗的资源比线程少很多，而且gorountine只是Go内部的概念，不会在操作系统层面有对应的实现。在Go里启动的各个gorountine之间是并行的，每个gorountine可能

会映射到一个系统线程，也可能多个gorountine共用一个线程，如果是多核的机器，不同的gorountine会自动分配到不同的核心。gorountine间的切换也由Go来控制，不需要程序员操心。gorountine占用的内存远小于系统线程或进程，gorountine间的切换成本也很低。程序里可以轻易创建数万个gorountine做并行，而不用担心会占用过多的系统资源。

Go语言利用gorountine实现并发，用chan实现消息通信。通过这两个概念的配合，提供了对并发的支持。

值得注意的是gorountine里对i的赋值操作符:=，这个操作符是指声明并创建一个变量，并赋初值。变量的类型会自动设置成初值的类型。Go继承了C语言的静态类型的特点，同时也在一定程度上借鉴了C++类型推导的特性（类似于C++ 0x的auto关键字，如果你知道C++ 0x的话）。另一种更传统的写法是：

```
var i int = nnn
```

传统的写法不仅多了不少字，而且还要自己注意类型是否匹配。所以Go更推荐使用:=（而且以后如果有Go语言混乱大赛，大概会用这东西来组成颜文字什么的XD）。

Node的核心函数Run（）

现在来看一下Node的核心函数Run（）。

```
func (n *Node) Run() {
  go func() {
    defer func() {
      if x := recover(); x != nil {
        println(n.name, "panic with value",
        x)
        panic(x)
      }
      println(n.name, “finished”);
    }()
    // Run函数的核心
    for n.in_degree > 0 {
      received := <- n.in_ch
      n.inode.receive(received)
      n.in_degree--
    }
    ret := n.inode.run()
    n.out_ch <- ret
  }()
}
```

进入Run后就启动了一个gorountine，保证每个Node节点间都是并行的。在gorountine的内部，先略过defer不看，看Run函数的核心部分。首先等待所有前驱节点工作的完成。关键字for是Go语言里的循环语句，一般来说有四种用法：

```
for {}  // 相当于C语言里的while 1 {}
for i := 0; i < xx; i++ {} // 相当于C语言里
  的for (int i=0; i<xx; i++) {}
for i > 0 {} // 相当于C语言里的while (i>0) {}
for index, item := range array {} // 相当
  于Python里的foreach，index是循环序号
```

这里用的是第三种用法。由于每个节点都是通过ConnectTo来和前驱关联在一起，因此in_degree的数值就是前驱的个数，当有前驱完成，由ConnectTo启动的gorountine就会把前驱的输出放入in_ch里（见前面对ConnectTo的分析）。for循环等待所有前驱节点的输出，并把输出传入inode的receive接口做处理。

之后调用inode接口的run进行节点自身的处理，并将处理后的返回值赋给ret。最后将ret的内容从out_ch里输出。

defer

defer是Go另一个很有意思的特性，借鉴自C++的析构函数和Java的final。defer指定的函数不会立刻执行，而是在当前函数退出时才执行。defer主要是用来做一些清扫类的工作，比如常见的关闭文件、释放缓存。这里的defer用来处理inode.run()在执行时可能出现的异常。

Go的异常机制也与其他语言不同。一般来讲，Go的错误处理类似常见的C函数，推荐使用返回值作为控制手段。但是在一些情况下，可以通过内建的panic函数来触发一个异常。如果这个异常不被捕获，就会引起程序真正panic。捕获异常使用内建的recover函数，如果这个函数执行前有panic发生，就会返回调用panic时传入的参数；如果没有panic发生，就返回一个nil——Go里的空指针。

defer里if的用法也很有意思。在if执行时，分号前的部分对变量x做初始化，分号后才是这个if的判断值。x的作用域限制在if的语句块里。这里Go借鉴了C++的思想：尽可能缩小变量的生命周期。当然，也可以使用传统的写法：

```
x := recover()
if x != nil { ... }
```

如果recover得到的值不为nil，就简单输出异常并重新抛出。如果一切正常，就打印一句提示并退出。

应用Node来构造各种节点

上面就是Node的实现。接下来就要展示如何应用Node来构造各种节点了。

DoubleNode

首先是DoubleNode，结构如下：

```
type DoubleNode struct {
  data int
}
```

对DoubleNode来说，只需要一个data存储需要处理的数值就可以，因此结构很简单。然后是Node的处理函数：

```
func (n *DoubleNode) receive(i int) {
}
func (n *DoubleNode) run() int {
  return n.data * 2
}
```

由于DoubleNode是初始节点，不会接收数据，所以receive没有做任何事情。run里将data的值翻倍并返回。

值得注意的是DoubleNode并没有从NodeInterface做继承，除了实现了NodeInterface的两个接口，甚至没有任何提到NodeInterface的地方。这是Go的interface与Java和C++侵入式的接口实现最大的不同。Go的interface并不需要实现类与interface有任何直接的关联，在编译时，编译器会自动检查一个类是否符合interface的要求，并在运行时做动态绑定。由于并不要求强制的继承，因此在设计类的时候也不会受到继承体系的限制，想让一个类符合某个interface，只要加入相应的函数实现就可以，不用改动整个继承体系。

之后是如何生成DoubleNode：

```
func NewDoubleNode(name string, data int)
*Node {
  return NewNode(name, &DoubleNode{data})
}
```

这里将新生成的DoubleNode实例的指针直接作为参数传入了NewNode，Go的编译器会帮你处理背后的工作。注意interface只能接收一个实例的指针，而不能直接接收实例作为参数。

SumNode

之后是SumNode的实现：

```
type SumNode struct {
  data int
}
func NewSumNode(name string) *Node {
  return NewNode(name, &SumNode{0})
}
func (n *SumNode) receive(i int) {
  n.data += i
}
func (n *SumNode) run() int {
  return n.data
}
```

SumNode在接收到其他Node传入的数后，会将其累加到自己的data里，最后只用简单传回data的值就完成了全部工作。其他函数很简单就不细说了。

Node组合

现在是将这些Node组合在一起完成工作的时候了：

```
func main() {
  sum := NewSumNode(“sum”)
  sum.Run()
  for _, num := range [5]int{1, 2, 3, 5,
6} {
    node := NewDoubleNode(“double”, num)
    node.ConnectTo(sum)
    node.Run()
  }
  println(<- sum.out_ch)
}
```

Go在编译可执行文件时，会自动调用内部定义的main函数。main函数里，[5]int{1, 2, 3, 5, 6}表示一个含有5个元素的int数组，“_”和Python里的“_”含义一样，是一个匿名变量，表示这里将接收一个值，但是程序后面会忽略这个值的具体内容，这里是忽略掉range返回的循环序数。其余的内容希望已经简单到读者能一眼看懂（看不懂的话，大概是Go语言的失败吧）。值得注意的是，sum的Run并没有阻塞主进程的运行，这正是gorountine的并发所达到的效果。

写到这里，本文的内容就全部结束了。要说的是，这篇文章仅仅展示了Go语言相比C/C++/Java最大的不同，并不是Go的全部内容。比如像字符串、数组、分片、包管理、闭包等内容完全没有涉及。有兴趣的读者可以到Go语言的官方网站http://golang.org查阅相关文档。Go的官方网站也提供了一个在线的Go编译环境，可以编译执行Go的代码，体验Go的魅力。本文程序例子的完整代码请见《程序员》官网（http://www.programmer.com.cn）。

作者简介 | 李兆海

曾做过一段时间的嵌入式程序，后赶着网络大潮的尾巴，转而关注网络后端技术。喜欢乱翻书，碰到有兴趣的都会拿来研读一番。常年保持兴奋的兴趣点有编程、数学、物理、古典音乐。

■ 责任编辑：董世晓（dongsx@csdn.net）

基于LIRS缓存替换算法的实践

■ 文 / 徐永睿

为了合理和高效地使用缓存，作者提出了一种基于LIRS的缓存替换算法，为上层业务开发提供了统一、简单的接口。

在各种规模的计算机系统中，许多Web应用都将数据保存到RDBMS中，应用服务器再从中读取数据。但随着数据量的增大和访问的集中，就会出现RDBMS负担加重、数据库响应恶化、网站显示延迟等重大性能问题。这时，通过对应用系统使用缓存就成为当务之急。在当前缓存中间件产品中，Memcached较为成熟，且在Wikipedia、Twitter等有着较为成功的运用。但由于Memcached采用的内置缓存替换算法是基于LRU（Least Recently Used）的，因此，在一些数据访问模式中缓存命中率不高、对于某些对缓存命中率要求特别高的项目，Memcached并非是一种最佳选择。我通过查阅相关资料，实现了一种基于LIRS的缓存置换引擎。该引擎封装了LIRS缓存替换算法，为上层业务开发提供了统一和简单的接口，使用起来相当容易。

LRU及LIRS算法原理

在当前大量使用的缓存中间件如Memcached中，通常采用LRU算法认定并淘汰不常用的数据。LRU的实现异常简单：一个队列，新的数据从队列尾部进入，当要淘汰过时数据时，选取队列头部的数据进行淘汰。为了更好地介绍LIRS，引入Recency的概念。Recency是指一个缓存数据到缓存队列尾部的距离。如图1所示，数据9到队列末尾距离为2，因此数据9的Recency为2。

在LRU中，通过引入Recency，就将算法的基本思路变换为当缓存容量达到缓存队列的最大值时，淘汰掉Recency值最高的那个数据，如图1中的数据6（Recency = 4）。

在LIRS中，也引入一个类似的概念IRR（Inter Reference Recency），指的是同一个缓存数据在两次访问间隔中，访问过的其他缓存数据的个数。在图1中，如果当前访问的数据是9，本次与上次对资源9的访问之间间隔了资源1和资源2的访问，所以资源9的IRR为2。引入IRR有什么作用呢？其实大家可能已经猜到了，在LIRS的假设中，如果一个资源访问间隔周期比较短，那么在未来这个资源有可能很快就需要访问到。

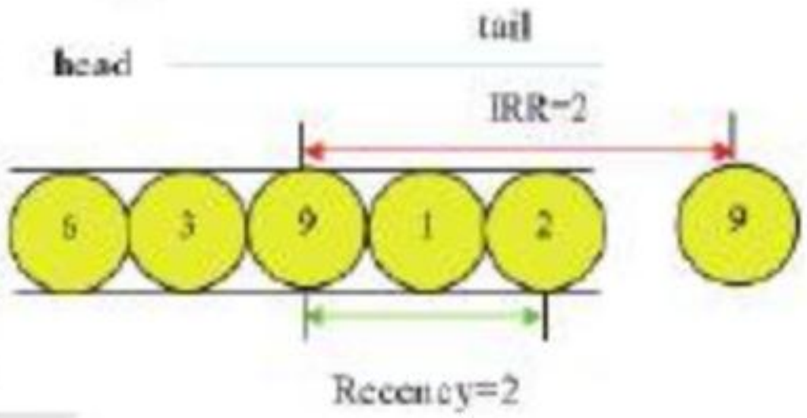


图1 Recency的引入

常用的LRU算法在考虑缓存替换时并没有考虑到对资源的其他相关访问，只是简单地认为最久都没有访问到的数据被继续访问的概率是最低的。而LIRS算法通过记录每一个缓存数据的访问间隔，将访问间隔最大的数据淘汰掉。相对于LRU，LIRS的假设明显更符合绝大多数的应用程序。在我开发的系统中，所有资源中大约有10%的资源属于访问频率特别高的资源（资源访问近似90-10分布），LIRS非常适用于这种数据访问模式。在缓存配置相同的情况下，缓存命中效率相对于LRU提升了1倍左右。

资源描述

对资源进行如图2所示的建模。每个资源分为以下两种状态。

HIR（High IRR）：意味着资源访问间隔较大，这种状态的资源不一定在缓存中，缓存中的HIR资源随时可能被淘汰。

LIR（Low IRR）：资源访问间隔较小的资源，这种状态的资源一定会存在于缓存中且不会被淘汰。

当HIR资源访问间隔小于当前访问间隔最

图2 资源建模

大的LIR资源时，该HIR资源状态将变为LIR，而访问间隔最大的LIR资源的状态将变为HIR。

LIRS算法Cache布局

LIRS算法的Cache布局如图3所示。在LIRS中，由于所有的资源分为HIR和LIR两种状态，从物理Cache中分配一个较大比例部分（比如99%，可以根据应用自行调整，在我的项目中当LIR Cache占总Cache容量99%时缓存命中率最高）缓存LIR资源集。因此，LIR资源集容量就等同于物理Cache分配给LIR资源集的容量。由于剩下的资源都属于HIR资源集，而物理Cache分配给HIR资源集的容量较小，因此，只有极少一部分HIR资源能够驻留在物理Cache中。当访问剩下的HIR资源时出现缓存缺失。

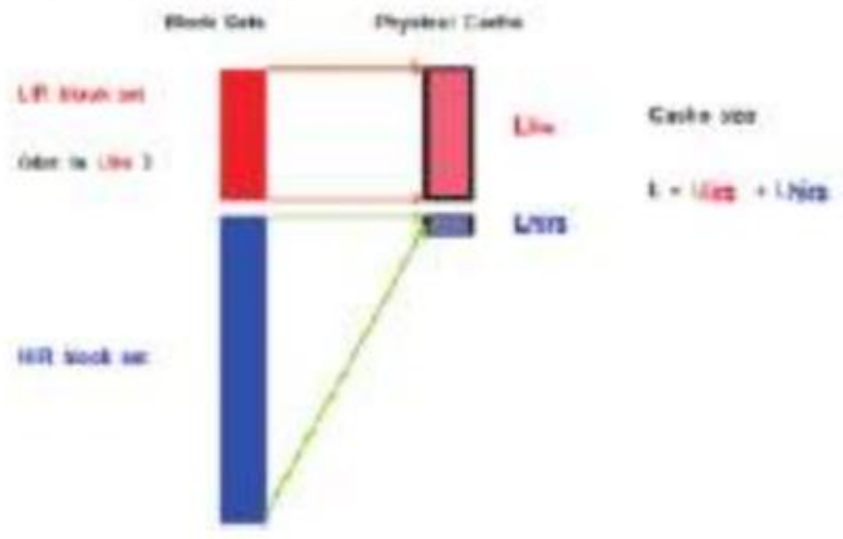


图3 LIRS算法Cache布局

LIRS算法数据结构

在LIRS的数据结构中，不同于LRU只包含唯一的一个缓存队列，由于资源分为两种不同的状态，因此，LIRS实现包含了两个不同的缓存队列。一个缓存队列仅用来存放HIR的资源（HIR队列），该队列所能容纳的HIR资源个数严格等于在物理Cache中分配给HIR资源集的大小所能容纳的资源个数（对应于图3中的Lhirs部分）。由于物理Cache分配给HIR资源集的Cache容量通常很小（比如1%），因此，该队列只能保存少量的HIR资源。另一个缓存队列则包含了全部的LIR资源以及部分HIR资源（LIR队列）。之所以LIR队列还会包含部分HIR资源，是因为队列中的这些HIR的资源IRR值至少已经低于LIR资源集中IRR值最高的那个资源。如果这些HIR资源被继续访问，则将替代IRR值最高的LIR资源而加入LIR资源集合。更进一步，在LIR队列的实现上队列头的资源一定是LIR资源（如果是HIR资源则其本身IRR值已经超过了当前LIR资源集中IRR值最高的那个资源，即便未来该资源被访问，也不能进行替换）。

在实现上，由图4可知，两个缓存队列直接使用C++ STL中的deque。相对于采用适配器版本的queue，deque提供了更丰富的接口操作，避免了vector在内存分配上的低效。

```
deque<Resource*> _LIRQueue;
deque<Resource*> _HIRQueue;
```

图4 LIR数据结构实现

算法描述

第一，所有的资源状态都初始化为HIR。当LIR资源集中现有资源个数小于Cache分配给LIR资源集大小时，新访问的资源状态直接转换为LIR，并进入LIR队列，直到LIR资源集中的资源个数等于物理Cache分配给LIR资源集的大小所能容纳的资源个数（之后LIR资源集中的资源个数将保持不变），如图5所示。

第二，当访问的资源属于LIR资源集中的资源时，此时缓存命中。只需简单地将LIR队列中的当前资源移到队列尾部。但一种特殊情况是，当前访问的LIR资源是LIR队列中队列头的资源，为了避免HIR资源出现在LIR队列头部，因此，此时必须执行一个队列修剪操作将队列头部的HIR资源移除队列，直到队列头部资源是一个LIR资源，如图6所示。

第三，若当前访问的资源是HIR资源并且该资源在缓存中，首先将该HIR资源加入LIR队列，然后查找该资源是否在LIR队列中已经存

图5　初始操作

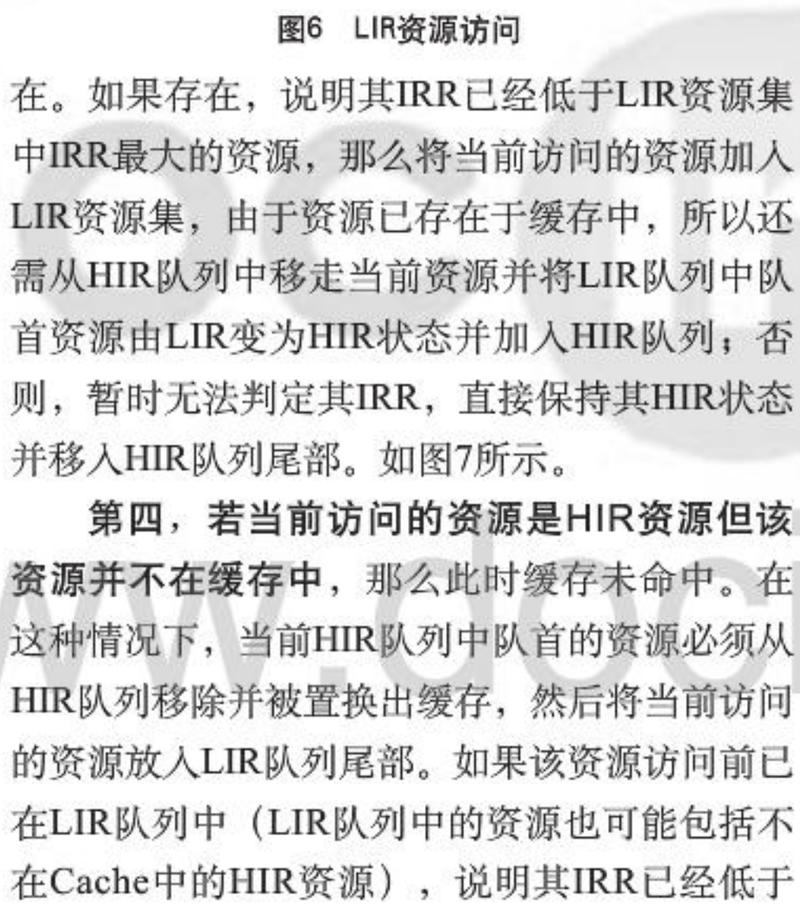


图6　LIR资源访问

在。如果存在，说明其IRR已经低于LIR资源集中IRR最大的资源，那么将当前访问的资源加入LIR资源集，由于资源已存在于缓存中，所以还需从HIR队列中移走当前资源并将LIR队列中队首资源由LIR变为HIR状态并加入HIR队列；否则，暂时无法判定其IRR，直接保持其HIR状态并移入HIR队列尾部。如图7所示。

第四，若当前访问的资源是HIR资源但该资源并不在缓存中，那么此时缓存未命中。在这种情况下，当前HIR队列中队首的资源必须从HIR队列移除并被置换出缓存，然后将当前访问的资源放入LIR队列尾部。如果该资源访问前已在LIR队列中（LIR队列中的资源也可能包括不在Cache中的HIR资源），说明其IRR已经低于LIR资源集中IRR最大的LIR资源，则将当前访问的资源加入LIR资源集，并将LIR队列中现有队首资源由LIR变为HIR状态并加入HIR队列，然后执行队列的修剪操作；否则，该资源保持HIR状态，放入HIR队列尾部。如图8所示。

图7　驻留Cache的HIR资源访问

图8　未驻留Cache的HIR资源访问

LIRS算法与LRU算法性能比较

为了准确全面地比较LIRS算法与LRU算法，首先将LIRS算法分成LIRS1和LIRS2两类算法。正如前面LIRS算法Cache布局中所述，HIR资源集在物理Cache中所占比例是可以调整的，因此，在LIRS1中，LIR和HIR资源集各占物理Cache容量的50%，而在LIRS2中，LIR资源集占物理Cache容量的99%。

资源个数指的是不同的资源，Cache容量是当前测试环境下设定的物理Cache能存放的不同资源的最大个数，而资源访问请求是模拟用户对资源的访问请求序列。由于我开发的系统中，所有资源中大约有10%的资源属于访问频率特别高的资源（资源访问近似90-10分布），因此，此处随机生成的资源请求访问序列也近似符合该假设。表1是具体缓存缺失率比较数据。

从表1可知，LIRS1相对于LIRS2来说，缺失次数较高，但整体性能表现仍然优于LRU算法。我认为主要原因是LIR资源能在缓存中驻留相对较长的时间，面临更少的缺失，从而更大的LIR资源集使得低IRR资源面临更少的缓存缺失。

在不同资源个数为1000、Cache容量占资源总容量10%、访问请求为3000、5000、10000次时，LIRS2缓存缺失次数只有LRU的59.9%、54.8%、51.6%。对于绝大多数应用系统而言，

表1　LIRS与LRU缓存缺失次数比较

缓存容量只占系统资源整体容量的极小一部分，而在这种前提下，随着应用系统访问次数增多，LIRS相对于同缓存配置下的LRU拥有更为理想的缓存性能。

随着缓存容量增大（此处为资源容量30%），LIRS与LRU性能几乎一致。这也比较好理解，在我的应用中，只有10%左右的热点资源，占总资源容量30%的缓存已经相对足够大，所以不同缓存替换算法表现几乎一致。如果假设一个极端条件，缓存容量等于资源总容量，那么将根本不会存在缓存替换，因此，任意的缓存替换算法都会有完全相同的性能表现。但在实际应用中，这几乎不可能成立，因此并不影响LIRS的优异性能表现。此外，当不同资源为2000而缓存容量保持100时，缓存容量只占资源总容量的5%，由于系统缓存总容量过小，即便是LIRS2缓存缺失率也从缓存容量占资源总容量10%时的13.86%升为55.81%（10000次访问请求）。此时，由于所有缓存算法缺失次数都在增加，LIRS相对于LRU优势与缓存容量占资源总容量10%时相比较有所减弱，但LIRS2缓存缺失率仍然分别只有LRU的95.3%、92.7%、91.5%。因此几乎所有类似基于资源概率访问的应用系统在采用LIRS缓存算法后，都能得到明显的性能提升。

LIRS实践中的问题及解决方法

看了性能对比数据，或许大家已经跃跃欲试了。但可能会有疑问：既然LIRS性能如此好，为何当前如Memcached之类的解决方案内部会采用LRU进行缓存置换？其实很重要的一个答案就是并发访问。对于任意的缓存置换算法，当发生缓存替换时，必须将内部数据结构进行锁定然后串行修改。当前为了解决缓存的并发访问问题，提升可扩展性通常都是牺牲缓存高命中率为代价的。以PostgreSQL为例，在其内部缓冲区的管理中，置换算法曾先后使用过LRU、2Q、ARC，当前最新版本9.0.1使用的也是Clock sweep算法。这些算法可以认为都是基于Clock思想的LRU变形方法。采用Clock思想的一个特点是当缓存访问命中时，设置访问标志或是增加资源引用计数，当需要进行置换时，替换掉那些访问标志没有被设置的或者是引用计数为0的资源。而所有资源通过哈希等手段被映射到Clock的某一片段（类似Dynamo中的一致性哈希环），这样当进行并发控制时，只会对Clock的局部进行锁定，锁粒度被极大降低从而提升了扩展性。但是，这种基于LRU的方法缓存命中率通常不高。

在LIRS的实践过程中，为了保持LIRS高缓存命中率的优势同时解决LIRS数据结构在修改时被锁定而导致的无法并行访问的问题，我采取了一种批量更新的方法：在每个客户端维护一个固定大小的FIFO队列用来记录每个客户端的资源访问情况，当客户端所访问的资源存在于缓存中时，将该次资源访问请求简单放入该客户端的FIFO队列中并从缓存中直接返回资源，此时无需更新LIRS的HIR和LIR队列。因此，在缓存命中情况下，由于LIRS数据结构只读，大量客户端可以并发访问。当某个客户端所访问的资源不存在于缓存中时，此时才锁定LIRS的数据结构，将该客户端FIFO队列中记录的资源按先后顺序在LIRS数据结构上进行缓存置换操作，最后清空该客户端FIFO队列。采用这种方法，缓存操作几乎不需要锁，具备很高的扩展性。即便当缓存未命中时，由于只需要加一次锁就能实现对LIRS数据结构批量更新，极大减少了加锁次数，避免了昂贵的锁代价。

在实现中，为了能尽快在LIRS的HIR和LIR队列中反映出客户端对资源的请求情况，客户端的FIFO队列设置固定大小，并设置一个队列更新阈值。只要客户端FIFO队列容量达到更新阈值时，客户端就尝试获得锁的所有权（TryLock），进行对LIRS数据结构的更新操作（TryLock性能代价很低）；如果不能获得锁的所有权，仍只需要简单将该资源请求放入FIFO队列，下次进行资源访问请求时再次尝试。而当到达队列固定大小时，客户端必须强制等待获取锁（Lock）进行更新（如图9、图10所示）。

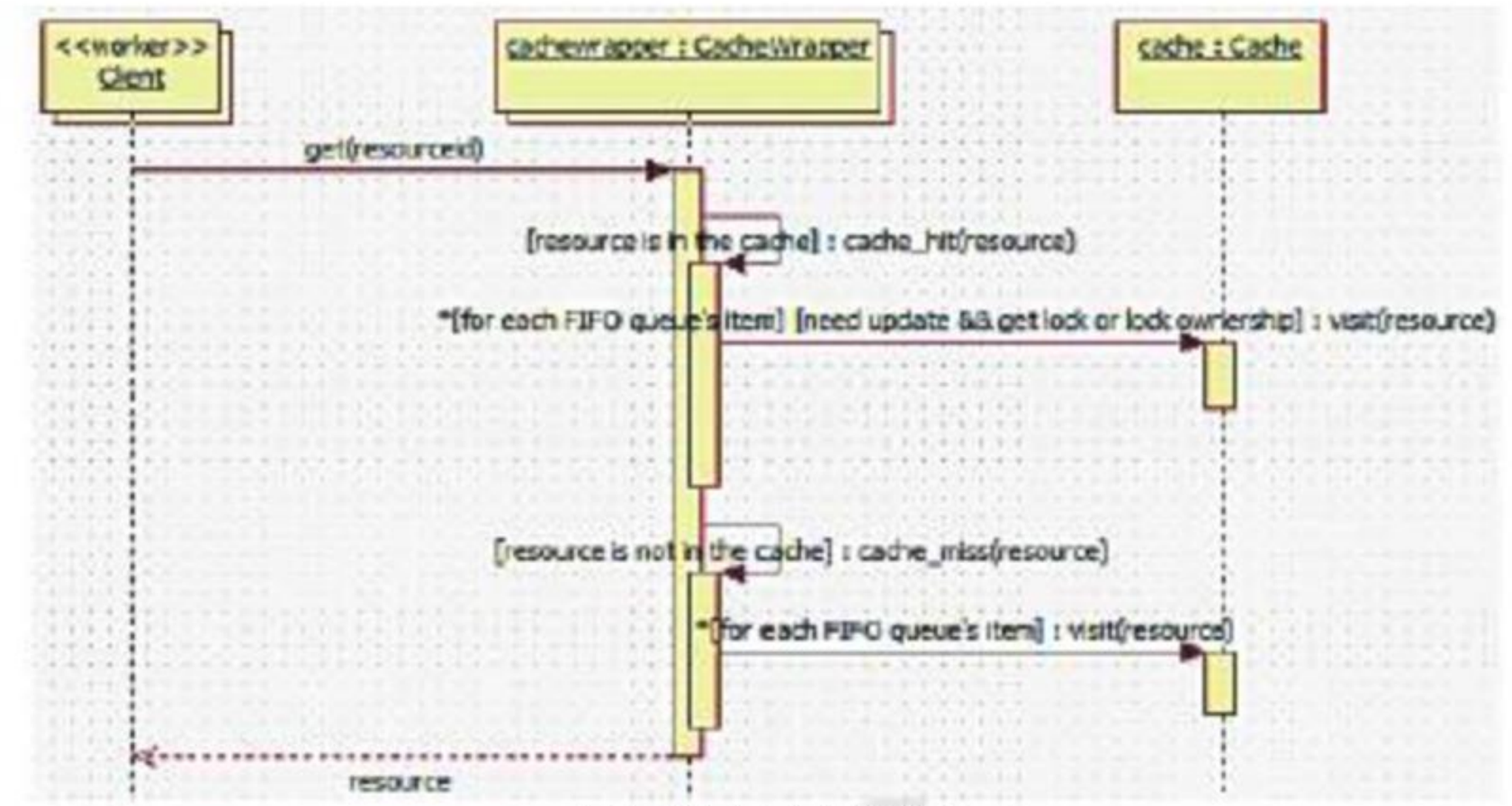


图11　缓存未命中时批量处理

```
const int S = 64;   //fifo队列大小
const int threshold = 32;   //fifo队列阈值
array<Resource*,S> Queue;   //fifo队列
int tail = 0;   //fifo队列索引
Cache* cache = new Cache(); //Lirs缓存算法封装对象

void cache_hit(Resource* r) {
    Queue[tail] = r;
    tail++;
    bool trylock = 0;
    if(tail >= threshold)   //队列达到阈值
        trylock = TryLock();   //尝试获取锁
    if(!trylock) {  //尝试失败
        if(tail < S)   //若小于队列容量，直接返回
            return;
        Lock(); //否则必须强制等待获得锁
    }
    //对fifo队列执行缓存置换操作
    for_each(Queue.begin(),Queue.end(),
        bind(&Cache::Visit,cache,_1));
    Unlock();
    tail = 0;
    Queue.clear();
}
```

图9　未驻留Cache的HIR资源访问

```
void cache_miss(Resource* r) {
    Lock(); //必须强制等待获得锁
    //对fifo队列已有资源执行缓存置换操作
    for_each(Queue.begin(),Queue.end(),
        bind(&Cache::Visit,cache,_1));
    cache->Visit(r);   //对当前缺失资源进行缓存操作
    Unlock();
    tail = 0;
    Queue.clear();
}
```

图10　缓存未命中时批量处理

用户接口和交互

对于一个需要被复用的软件构件来说，简单易用的用户接口、构件内部组件松散耦合的交互显得尤为重要。在实践中，我将LIRS算法本身封装到Cache中，而将解决算法并发访问的逻辑封装到了CacheWrapper中。当在实践中发现更优秀的用于解决并发访问的方法时，可以简单提取出CacheWrapper接口并通过子类继承重构方法封装新的代码逻辑，这样对用户完全透明（要实现完全透明，需要采用工厂辅助类提供创建方法）而又不影响LIRS算法本身实现。同理，现有的并发解决方法也可以轻松地运用到LRU、2Q等其他缓存算法中。在用户交互上，Cache本身对用户透明，用户只需要与CacheWrapper的get方法进行交互，传入一个资源的ID即可，极大地简化了用户操作。如图11所示。

总结

LIRS算法在绝大多数应用环境下拥有更加理想的缓存命中率，更适合作为缓存系统的置换算法。通过采用批量更新的思想，能较好地解决LIRS的并发更新问题，使其能够得到更加广泛的应用。同时，在LIRS的实现过程中，代码也存在着一定程度的坏味，需要在今后进一步重构完善。

作者简介 | 徐永睿

系统架构设计师、系统分析师、软件设计师。现在西南科技大学从事并行计算、分布式存储方面的工作。关注函数式编程语言，经常参与Topcoder算法比赛。

■ 责任编辑：董世晓（dongsx@csdn.net）

淘宝Tair开源分析实践

■ 文 / 高洪 郭斌

本文通过试验以及源码的研究分析，详细展示了淘宝Tair结构数据存储系统的架构。

淘宝在2010年6月底开源了名为Tair的结构数据存储系统，而且它已经把这款产品立竿见影地应用在了用户登录、商品详情展示及淘江湖等重要环节上面。作为这样一款商用产品，必然值得我们去探究其精华。虽然从目前淘宝公布出来的资料里我们只能管中窥豹，所幸仍可以通过试验以及对其源码的研究分析，得出较为深入的客观结论。

Tair是一个怎样的系统

这又是一个基于Key-Value的分布式存储实现，尽管有人评论说相对于Google和Amazon的论文中提到的技术来讲，Tair的实现较为单薄，但我们认为从实践的角度来说，Tair的方案还是非常务实有效的。

Tair集群包括三个部分：处理并存储数据的Data Server、负责配置信息的保存和同步的Config Server，以及部署在应用端的Client。如图1所示。

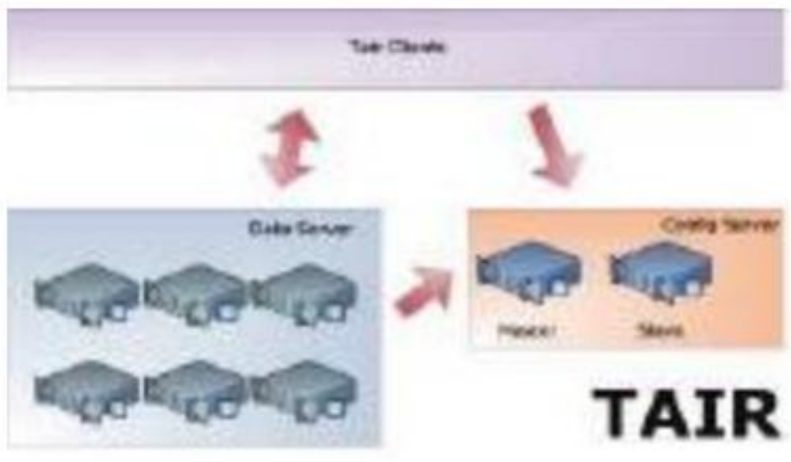


图1 Tair集群架构

Config Server只是一个轻量级的中心节点，仅负责配置初始化及同步。Config Server在实际应用中要部署两台，以传统的Master-Slave主备倒换方式进行工作。不过，如果Config Server采用主备方式的话，就无法与Data Server部署在一起了。

数据分布模型

Tair数据的分布依靠Config Server中维护的一张对照表。对照表中的行数越多意味着数据可以分布得更零散、更均匀，列数越多意味着数据的副本越多。这看似与Dynamo论文中Consistent Hashing的那个环大相径庭，但实际上道理是一样的，而且相比较而言更容易理解和实现。

对照表第一列是行号，也对应着Key的Hash值按总行数取模的结果（一行也就是Hash的一个桶）。第二列称为主节点，后面的列称为辅节点。如下表所示。

行号	主节点	辅节点1	辅节点2
0	192.168.10.1	192.168.10.3	192.168.10.4
1	192.168.10.2	192.168.10.4	192.168.10.1
2	192.168.10.3	192.168.10.1	192.168.10.2
3	192.168.10.4	192.168.10.2	192.168.10.3
4	192.168.10.1	192.168.10.3	192.168.10.4
5	192.168.10.2	192.168.10.4	192.168.10.1

Client要发起数据读写请求时，根据Key的Hash按行数取模定位到某一行，也就找到了负责这个Key的节点列表，进而执行数据存取操作。当新节点加入或有节点长期故障需要涉及数据重分布时，将数据进行迁移后修改对照表中对应位置的IP即可。总之，对照表确定后，数据的分布也就确定了。

Tair设计这张对照表还有容灾的考虑。在Config Server填这张对照表的时候，通过以IP掩码的方法支持机架和数据中心感知机制，使对照表中同一行的几台节点分别落到不同的机架和数据中心，充分体现了Tair的分布性。

这种方式有一个小问题是，因为对照表行数必须是一开始就确定的（推荐值是1023行），如果在一个应用实例中，最初只有几台节点，而对照表行数非常多，则每台节点上负责的行就会很多，而这是要耗费大量内存资源的。反之，如果最初设定的对照表行数较少，那么假如这个应用实例的系统大量扩容（比如扩容到节点数超过对照表行数），可能无法将数据均匀分布。所以一个应用必须对系统未来的扩容情况进行预估来确定最合适的参数设置。Dynamo论文中Consistent Hashing那种环的方式则不存在这个问题，因为可以根据系统规模进行“裂变”，不过那种裂变也是有代价的，因为数据迁移时必然需要将一份数据根据规则分成两份，实现复杂度也较高。

另外一个问题是，一套Tair系统支持的数据存储模型是全局配置的，包括数据副本数、存硬盘还是内存等这些参数。而在实际使用中，不同应用对数据的可靠性和可用性要求是不同的，比如一种应用对数据访问性能要求特别高但可以没有冗余副本，而另一种应用对数据访问性能没有要求但一定要可靠，则必须搭建两套环境才能满足。建议Tair今后能支持在一套系统中为不同数据提供不同的存储方式，以提高系统配置的灵活性，节省维护成本。

Data Server结构及存储模型

Tair的Data Server结构如图2所示。

- Tair服务运行框架负责整个服务的初始化、启动、停止、请求处理管理、链路管理。
- Tair服务管理提供运行期间各种服务的管理，包括：存储管理、数据迁移管理、数据复制管理、数据导出管理、服务器配置表管理。
- Tair存储管理处于Data Server的核心位置，包括内存管理和文件管理等。

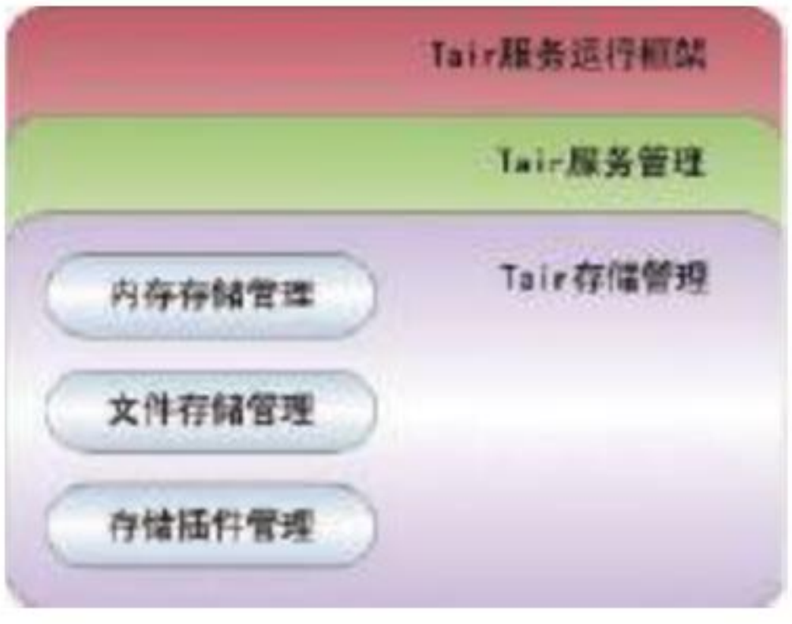


图2 Data Server的结构

存储管理又包括了内存存储引擎和文件存储引擎（持久化），并支持其他存储插件。内存存储部分与Memcached类似，不同尺寸数据分别管理，整片申请内存。有数据写入时，把同一尺寸数据放到对应尺寸的内存片中；文件存储部分采用索引文件结合共享内存存储的方式，便于快速检索数据文件中的数据。也可以使用BerkeleyDB、Tokyo Cabinet、MySQL等替换掉Tair原生的存储引擎，只需做一个插件根据Tair的接口进行适配即可。这也是Tair的一个特色，因为对不同应用来说，其应用特征不同，这个插件机制能够让应用找到最适合的存储方式。

由于内存是按照数据的长度分类管理的，所以为了保证内存片的利用率，Tair设置了内存片整理线程，每隔几小时就根据LRU（Least Recently Used，最近最少使用算法）情况进行内存片的整理操作，释放出一定量的内存片，然后对内存需求比较多的长度类别可以获得较多的分片，最终内存能够实现按需分配。

自动数据重分布

数据的自动重分布也是Tair的一个亮点。Config Server会检测节点的加入和故障，自动重构对照表。数据节点发现对照表变更时，会根据新的数据分布规则进行数据迁移。迁移过程中，仍由迁移的源节点提供服务，同时源节点会不断把数据同步到目的节点上。迁移完成后，客户端将从Config Server获取新的对照表。

但是根据测试情况来看，在增加节点后的迁移过程中有少量写失败的情况。究其原因，在于为新的对照表生效之前，源节点上有500毫秒的时间是不接受写和删除请求的。基于Tair的架构，这样做是比较合理的。因为在这个时间段内，假设源节点接受了写请求并自己处理，迁移过程可能永远也结束不了；假设源节点接受了写请求并转给目的节点处理，这时目的节点可能还不具备提供服务的能力。对Tair的应用来说，天生是要对各种失败进行感知和处理的，所以迁移过程中有少量失败应该也可以接受。

数据一致性和可用性

数据一致性一般关注两方面，一方面是分布在各节点上的数据，在读取时如何得到正确的版本；另一方面是并发写入时，以哪个版本为准。

对第一方面，Tair的实现中没有用到Dynamo那种Quorum NRW的策略，而是由对照表每一行上的节点按主次优先级对外提供服务。写入数据时，主节点写入成功后，会将数据向后面各辅节点发送，不等辅节点响应，立即向Client返回结果。仅当主节点可用时写入是成功的，且主节点故障时Config Server会变更对照表。读取数据时，如果主节点正常则只从主节点读取，否则依次轮询辅节点读取。

这种做法效率会比较高，但一致性和可用性是个问题，首先单点故障后数据的写入会有一段时间是失败的；其次，如果单点故障相继发生，会造成同一个Key在各节点上的数据不一致的情况。打个比方，辅节点1故障时写了Key为abc的数据，这时是写入了主节点和辅节点2。当辅节点1恢复后，如果某时刻主节点又出现故障，这时有个Key为abc读请求过来，会去从辅节点1上读取数据，而辅节点1上是没有abc这个数据的。

也就是说，Tair系统在单点故障时，在可用性和数据一致性方面的表现还较为脆弱。另外，Tair似乎也没有数据不一致的发现和修正机制，即上述例子中的abc在辅节点1上一直是没有数据的。

对第二方面，虽然Tair的版本号是一维的而不是向量的，但是引入了一个“基于的版本号”机制，解决了数据并发写操作中的一致性问题。即Client发起写操作时，会先发起读操作读到数据当前的版本，然后本次写操作就“基于”这个当前的版本。

如果并发有两个Client同时发来的写操作，读到的版本都是10，然后往主节点上发送写请求时，都是基于版本10。因为同一个节点上收到的请求总是有先后的，那么先到的那个操作会得到执行，然后版本号立即改成11，轮到后面那个处理时，因为基于的版本10已经不是当前版本了，所以主节点拒绝Client的这个请求，由应用决定是重做还是放弃。因为Tair只许在主节点上完成写入，所以在这个前提下Tair的这套策略还是无懈可击的。

总结

在著名的分布式系统CAP理论中，Consistency、Availability以及Tolerance of network Partition不可兼得。那么在淘宝的Tair中，可以看出一切都是围绕着Partition来设计的，其机架感知和多数据中心容灾的支持具有绝对优势，当然这个优势是以牺牲可用性和一致性为代价的。在保证了Partition的前提下，Consistency和Availability互相妥协，特别是Availability这一方面妥协得更多。

我们看到在Tair官方性能测试报告的表格中有“成功率”一栏，数据都在99%以上，经实际测试，报的最多的错误是“duplicate busy”，即复制繁忙。对Tair的应用来说，充分考虑Tair返回各种失败的场景几乎是必需的。这也体现了Tair的设计哲学，要求客户端对服务端的各种疑难杂症有足够的容忍。

不过任何系统都是为了解决特定问题而设计的，不可能存在万能的分布式系统。总体来说，作为一个商用的开源Key-Value存储，Tair的很多设计思路和观点都很有学习价值和借鉴意义，比如数据的自动重分布的设计思想，以及灵活可定制的存储插件机制方面，但在可用性等方面的弱势制约了其应用范围。如果Tair能够在这些方面加以改进，必定是一款相当具有竞争力的产品。

作者简介 | 高洪　郭斌

高洪，中兴通讯业务研究院平台研发总工，中兴通讯技术专家委员会专家。研究方向为IN技术、信令网络、开放业务平台、IMS业务，目前主要从事云计算平台研发工作。

郭斌，中兴通讯软件工程师，从事云缓存项目的研发和管理工作，专注于云计算及下一代网络通信领域，热衷于业界前沿技术的学习研究及实践。

■ 责任编辑：董世晓（dongsx@csdn.net）

大众点评网的LBS应用之道

■ 文 / 王宏

LBS是什么？作者首先阐述了他所理解的LBS，然后分享了大众点评网在LBS方面的应用成果。

LBS应用在2010年得到了爆炸式的发展，大众点评网也是赶上了这波浪潮。在这篇文章中，我先简单谈谈个人对LBS的理解，然后介绍一下大众点评网在LBS方面的应用。

解读LBS

LBS全称Location-Based Service，从字面上理解，LBS即基于地理位置的服务；从应用层面上理解，LBS是基于用户所处的地址位置，提供的一种增值业务；而从技术层面上理解，我更习惯于将Location、Based、Service这三个单词拆开来解释。

Location：位置定位。随着科技的不断发展，现在的定位方法开始多种多样，例如GPS、AGPS、Wi-Fi、Bluetooth、RFID等。而其中最常用、成本最低的就要算GPS + AGPS + Wi-Fi了。为什么要将这三者结合呢？其原因在于它们的使用场景不同。GPS准确度高，但是初次定位时间较长，并且需要在开阔的地方，才能接收到卫星信号；AGPS是一种结合移动网络基站定位的辅助定位系统，优点是定位速度快、能够在室内定位，缺点是定位精度相对要差一些；而Wi-Fi定位则是随着移动终端配备无线网络而兴起的一种新型定位技术，其原理并不复杂，主要是利用城市中相对固定的无线路由器数据库进行辅助定位。在不同的场景下利用好各种定位技术，将能更准确地为用户提供位置信息。

Based：这个单词本身并没有特殊的含义，在这里将它拆解开，完全是为了解释清楚位置服务中一个重要的环节。有了经度、纬度的位置信息，我们需要将这些数字转换为用户可以理解的形式，如地图、地址等信息。地图不难理解，就是将位置标注在地图上，让用户

图1　大众点评网iPhone及Android的LBS客户端

直观地了解当前所在位置；而地址就是将经度、纬度信息转换为所在位置的国家、城市、街道，甚至是门牌号码，在不方便使用地图的情形下使用户快速了解所处位置。另外还有一个比较重要的问题就是偏移纠正。众所周知，由于各种定位手段的偏差以及一些法律、法规的问题，使得我们获得的位置信息可能与实际地理位置的信息存在偏移，如何处理这一问题也是影响位置服务好坏的重要因素之一。

Service：现在又要回到应用层面。有了位置信息，如何基于此为用户提供更多、更好的增值服务？目前形式也是多种多样，例如结合电子地图的导航，几乎所有开车的朋友都

已经在使用；社交、游戏网络，通过位置结合虚拟与实际，满足用户社交、游戏等方面的需求，这在国外已经有成功的案例——Foursquare和 Gowalla，其新颖的社交方式也受到广大用户的喜爱；还有就是为公众生活提供必备的衣、食、住、行等信息服务，用户可以方便地通过移动终端了解所在位置周边的餐饮、休闲、娱乐等本地化消费信息。生活服务信息的应用场景很多，逛街、商务、旅游等都会有这方面的需求，很可能是LBS应用爆发的突破点。

LBS在大众点评网的应用

大众点评网作为一家城市生活消费指南网站，主要致力于为中国消费者提供对本地餐饮、休闲、娱乐等生活服务发表评论、分享信息的平台，并为广大潜在的消费者提供客观、准确的本地化消费信息指南。目前，大众点评网首创的第三方点评模式吸引了千万网友的积极参与，由用户点评的包括餐饮、休闲、娱乐等100余万家生活服务商户已覆盖全国2000多个城市，且信息量和覆盖范围在不断地快速增长和自主更新中。其中，餐饮点评是发展最早也是目前消费者最为喜爱并聚集信息量最多的内容之一。

早在2005年年初，我们就意识到商户的位置信息对用户的重要性，并开始大规模收集商户的位置信息，利用电子地图技术直观地向用户展示商户的位置。近几年随着智能手机的普及以及移动互联网的高速发展，越来越多的用户需要在移动的环境下查询本地生活信息。而生活信息又是大众点评网的本行，所以在2009年，我们开始开发手机客户端上的应用，经过两年时间的研发，从最初的1.0版到现在4.0版（见图1），目前大约已经有了170万的装机量（见图2）。

图2　大众点评网用户签到的热点图（截至2010年11月）

目前的版本涵盖了周边搜索、优惠券搜索、签到等功能。周边搜索能够帮你找到身边的美食、咖啡厅、停车场、加油站等常用信息，当然最有用的要算附近优惠券功能，它能立即告诉你周边有优惠的那些商户，在这个高CPI的时期，能帮你节省不少费用；不论是在餐馆无聊地等座位，还是想分享发现了一道好菜的兴奋心情，都可以通过签到功能，将你的心情分享给密友，甚至是同步到各大SNS网站，让大家知道。而一些特别的签到还可能获得商家更大幅度的折扣。

一年多的移动应用研发，也让我们在技术上获得了一些经验。最初的版本使用了常见的XML、JSON数据格式，用于客户端和服务器之间的交互，而这种方式在各种网络环境和移动终端硬件环境下表现不佳。用户有时使用Wi-Fi，有时使用移动网络，这导致数据传输的速度不同，加之各种手机的CPU处理能力也不尽相同，使得解析XML或者JSON的速度差异较大，从而大大影响了用户体验。

在2.0版本中，我们开始对数据进行压缩，使得数据更精简，以减少网络传输量，再利用轻量的序列化方案，减少客户端CPU的处理时间，让整个数据传输提高了2~5倍。

另外比较让人头疼的就是定位偏移的问题，由于相关政策的原因，公众实际获得的位置信息，在国内的地图上显示是偏移的，这会让用户感到疑惑，进而放弃使用。对这个问题各家公司有自己的解决方案，比较常见的是根据不同的城市设置不同偏移量，将定位获得的经纬度加上偏移量进行计算，在地图上标注用户及商户的位置，以减少这方面问题给用户带来的困扰。

相信未来几年，LBS将会是中国互联网的热点，我们也会利用LBS，研发出更强大、更简便的应用，使大家的生活更加便利、更加有品质。

作者简介 | 王宏

大众点评网系统架构师。多年互联网开发经验，对互联网的前端技术、系统平台架构等方向有丰富经验，喜欢Coding和Troubleshoot。近期专注于移动应用平台的设计开发。

■ 责任编辑：董世晓（dongsx@csdn.net）

趣谈CLR4的调试模型重构（下）

主持人：张银奎

《软件调试》一书作者，从事软件开发和研究10余年，对IA-32架构、操作系统内核、虚拟技术，尤其对软件调试有较深入的研究。翻译（合译）作品包括《数据挖掘原理》、《机器学习》、《人工智能：复杂问题求解的结构和策略》、《观止——微软创建NT和未来的夺命狂奔》》等。

红红的鞭炮摆上了货架，人们脸上堆满笑容，过年的气氛越来越浓了。长大以后不再有孩提时代那种盼望过年的急切心情，但至少春节长假还是很令人期待的。除了春节的例行节目，酝酿很久的新书也会在这个假期正式开工。新书写什么、书名还没有完全确定，但肯定是关于调试的，而且是软件的调试，别的不懂。为什么选择春节动工呢，迷信点说：日子好。

闲话就此打住，本期接上期继续谈CLR4的调试模型重构，上期谈的基本都是旧模型的缺点，其实还没有全部说完，为什么说这么多呢？主要目的是为了讲清重构的原因和目标。重构不是小修小补，而是大删大改。软件工程不是请客吃饭，对于.NET这样一种已经发布多年的开发技术，大删大改意味着投入大批人力物力。

事实上，.NET技术推出不久，就有人提出调试模型中存在的根本不足，但在CLR4之前，大家都表现出非常顽强的忍耐精神，不愿做大的改变。重构是需要决心和敢于冒风险的勇气的，重构不好怎么办？下面就来看看CLR4是如何处理这个烫手山芋的。

规范调试关系

在软件社会中，调试是很严肃的一件事情。在被调试进程中，原本的调试模型起一个辅助调试线程与进程外的调试器以普通的进程间通信方式对话的作用，这与正式的调试关系相比，简直就是“私通”，得不到操作系统的认可和保护，不够鲁棒，而且四处碰壁。比如，当所处的机器上启用了内核调试时，也会受影响，Visual Studio会弹出图1所示的对话框。点击其中的帮助按钮得到的解决方法居然是禁止内核调试，对于我来说，这简直无法接受。也许有人可以找借口说：“有哪个做.NET开发的程序员需要用内核调试呢？”的确比较少，但是做内核调试的人的确有时会做一些托管调试，比如我。

图1　托管调试与内核调试冲突（重构前）

为什么会有这个冲突呢？这得从内核分发异常的规则说起。内核函数KiDispatchException是Windows操作系统中分发异常的枢纽，对于来自应用程序的异常，比如调试器设置的断点所触发的断点异常，这个函数的中有类似如下判断：

```
if ((KiDebugRoutine != NULL) &&
  (PsGetCurrentProcess()->DebugPort == 0
    || KdIsThisAKdTrap(Tf, &Context)) )
{
  KiDebugRoutine(); // 分发给内核调试器
}
```

简单理解，上面代码的逻辑是如果当前系统启用了内核调试而且当前进程的调试端口（DebugPort）为空，那么就将当前的异常分发给内核调试器。

上一期我们介绍过，Windows用户态本地调

试是依赖调试端口的，当前进程的调试端口为空意味着这个进程没有被任何用户态调试器所调试。事实上，这个地方包含了对用户态调试的特殊照顾，如果当前进程在做用户态调试，那么内核就会把异常留给用户态调试器，不会走这个分支。因为重构前的调试模型没有使用调试端口，虽然也在做调试的事情，设置断点，触发异常，但内核分发异常时，并不知道这个进程在被调试，所以会执行上面的分支，执行KiDebugRoutine。对于普通的异常，比如throw抛出的软件异常，KiDebugRoutine一般会返回FALSE，让系统继续分析，但是对于断点这样专门用于调试的异常，KiDebugRoutine会将其分给内核调试引擎，内核调试引擎得到异常后会冻结整个系统，以便将系统中断到内核调试器。

图1所示的对话框正是为了避免这种尴尬局面而做的一个很负责任的措施，如果不这样做的话，开始调试后，说不定什么时候，可能突然整个系统就会断到内核调试器里，戛然而止了。

不小心又介绍了一个旧模型的缺点，这个缺点也是因为旧模型的调试关系不够“正统”所导致的。这种不规范的调试关系是旧模型的“病根”，也是重构时的首要“整改”目标。如何改呢？其实方法很简单，复用Windows系统中的现成用户态调试设施就可以了，这套设施扎根内核，在系统中享有诸多特殊照顾，依赖内核中的用户态调试子系统（DbgK）来协调调试事件，而不是普通的进程间通信。调试器使用调试端口（DebugPort）与被调试进程建立调试会话，这一关系登记在进程的核心数据结构中，名正言顺。图2画出了重构后的托管模型示例图。

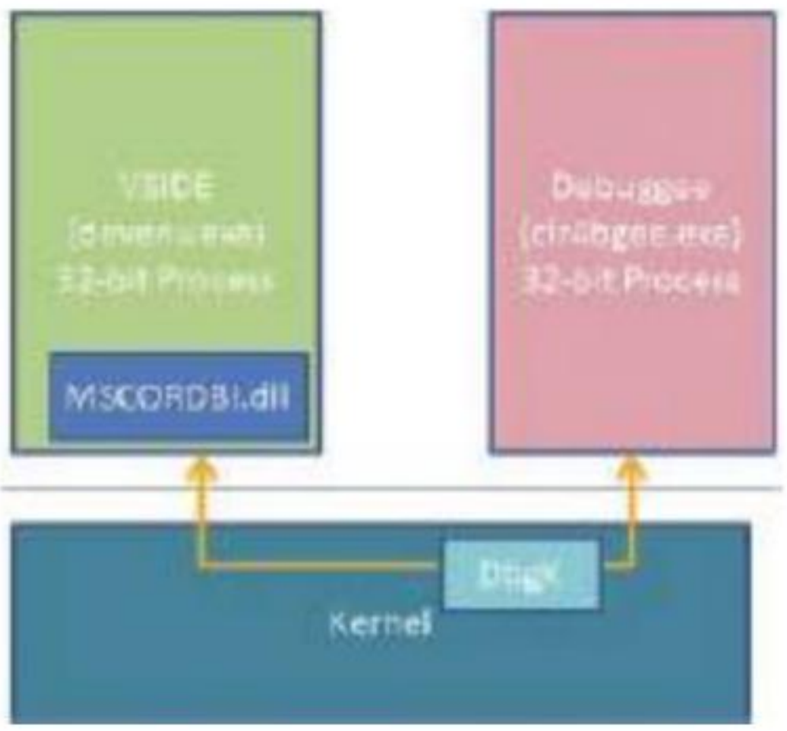


图2　重构后的托管调试模型

图2中左侧是调试器进程，以Visual Studio的IDE集成调试器（VSIDE）为例，右侧是使用.NET 4.0（CLR4）的被调试进程。二者使用Windows系统的本地用户态调试设施建立起正式的调试关系。正如这次重构的主要参与者之一Rick Byers在他的名为“ICorDebug re-architecture in CLR 4.0”的博客文章中所说的：

“Under the hood we're built on the native debugging pipeline.”

新的调试关系建立后，便不再允许以前的“二调一”出现，例如，当VSIDE已经开始调试某个托管进程后，如果再尝试用WinDBG附加到这个托管进程，那么便会得到图3所示的“拒绝”对话框。

图3　重构后不再允许“第三者”

仔细读了图3中的解释信息之后，我不禁笑出声来。这段话是对错误代码“0xc0000048”的说明，在WinDBG中执行“!error 0xc0000048”可以得到一样的信息，其中的三个attempt说了三种情况，都是企图“占领”某个端口，但是这个端口已经有“人”了。这不禁让我想起一个反调试软件，它启动两个进程实例，一个调试另一个，自己先把调试端口占了。

改写调试器接口模块

ICorDebug是托管调试的核心接口，它公开了一组方法向上提供调试功能，以简化调试器的开发。这一接口是实现在名为“mscordbi.dll”的动态链接库中。在重构之前，这个接

口的实现思路主要是与位于被调试进程中的调试辅助线程通信。如上一期所说的，位于被调试进程中的部分通常称为左端（LS），位于调试器进程中的部分称为右端（RS），而位于“mscordbi.dll”中的代码是右端的核心。在SSCLI的源代码中可以看到它们的源代码，例如，下面就是执行以附加调试动作的DebugActiveProcess函数的主要代码：

```
HRESULT CordbProcess::DebugActiveProcess(
    Cordb * pCordb,
    DWORD pid,
    BOOL win32Attach
)
{
    _ASSERTE(pCordb != NULL);

    HRESULT hr = S_OK;
    CordbWin32EventThread * pW32 = NULL;
     hr = CreateAndStartWin32ET(pCordb,
       &pW32);
    if (FAILED(hr))
    //
    hr = pW32->SendDebugActiveProcessEvent
     (pid,
       win32Attach == TRUE,
       NULL);
    // …
}
```

上面代码做了两个主要动作：一个是创建用于和左端通信的事件线程（Event Thread），另一个是向左端发送附加调试的命令（事件）。左右端通信建立，那么“调试会话”便建立了。

重构之后，这里显然需要修改，因为要调用操作系统的本地用户态调试API建立真正的调试会话。例如，图4所示的栈回溯记录的就是修改后的mscordbi事件线程在调用系统的DebugActiveProcess调试API，不再像以前那样只是把命令传达给左端就完事了。

图4　mscordbi在调用调试API和内核服务

改写数据访问模块

“Mscordacwks.dll”是CLR运行时中专门用于支持调试的模块，其全称为Microsoft .NET External Data Access Support，意思是提供对托管世界（数据）外部访问支持，WinDBG中的SOS扩展就是基于这个模块的。与mscordbi类似，该模块在重构前的实现思路也是与调试辅助线程通信，这导致了上一期提到的转储分析难题。例如，图5所示的栈回溯是当使用WinDBG+SOS来分析.NET 2进程的转储文件时，mscordacwks模块找不到调试辅助线程而抛出异常报告错误。

图5　数据访问模块抛出异常（重构前）

如何改进呢？就是不要依赖调试辅助线程，让数据访问模块具有从内存中分析托管世界的能力。CLR4中的新版本“mscordacwks.dll”已经实现了这一目标，下面是Rick在博客中简单提及的实现思路：

The way we attempt to address the maintenance issue is to build both the normal in-process CLR code (mscorwks.dll) and the out-of-process code (mscordacwks.dll) from the same code base. We use some fancy C++ templates to instrument all pointer dereferences so that we can read memory from the target process instead of the process in which the code is executing (“host process”). ——Rick Byers

这个实现思路达到了非常好的效果，让我们在分析转储文件时也可以得到与调试活动目标时一样精确的栈回溯。例如，图6显示的就是使用WinDBG+SOS分析.NET 4.0程序的转储文件时的场景，mscordacwks正在使用新的ClrDataStackWalk类根据转储文件产生包含托管信息的栈回溯，产生的结果如图7所示。

仔细看一下图7中的信息，再与上一期中的图9做个比较，可以说这两者间的差距非常悬殊，重构前对于托管函数对应的栈帧，只能看到一个二进制的地址，要想追溯出方法名不是件容易的事。

重构后，整个栈回溯非常清晰完整，从程序的起点Main函数，到窗口（Forms）过程，

图6　重构后的数据访问模块在分析转储文件时回溯栈

图7　转储分析时观察栈回溯

再到鼠标事件和按钮的响应函数（BtnDiv0_Click）。看了这样的栈回溯，我们很容易就知道这个线程的执行经过，让转储分析变得轻松愉快。

值得说明的是，要达到图7这样的效果，除了数据访问模块要具有解析托管世界的能力外，还需要在产生转储文件时，把内存中的“托管世界”保存到转储文件中。如果转储全部内存（MiniDumpWithFullMemory），那么不会遗漏托管数据。否则的话就需要特别注意把托管数据转储到文件中。不过这个工作在.NET 2.0时就做好了，就是在负责产生转储文件的模块“dbghelp.dll”（产生转储的API MiniDumpWriteDump就是这个模块导出的）中增加用于转储托管数据的功能，比如GenClrDataTarget和GenClrDataEnumMemoryRegionsCallback等C++类。因此，与Windows 7一起发布的“dbghelp.dll (6.1.7600.16385)”就已经包含了这些支持。

在软件领域，转储文件就像现实生活中的数码照片那样越来越流行。Windows 7甚至在任务管理器中增加了生成转储文件的功能，想给哪个进程拍照，只要一点右键选择“Create Dump File…”就可以“来一张”。为什么如此呢？因为转储文件对分析软件故障，特别是发生在客户机器上的产品期问题，来说太重要了。从需求角度来说，更好地支持转储分析是这次重构的主要目标。通过几个月的使用体验，我觉得这个目标实现得非常好。

调试辅助线程还在

正像Rick Byers在博客中所说的，托管模型的重构工作并没有在CLR4中全部完成，完成的部分主要是我们介绍的调试结构改变（把本来基于进程间通信的方式改为使用本地调试端口）和转储分析支持。对于调试活动目标（交互式调试），重构工作还没有完成，仍然离不开调试辅助线程的帮助。因此，当我们观察.NET 4.0的托管进程时，仍然可以看到调试辅助线程的“身影”，通常是1号线程，在执行“clr !DebuggerRCThread::MainLoop”。

当然，保留调试辅助线程的另一个原因是兼容老版本的调试器，以便可以用CLR4以前的调试器来调试CLR4程序，也就是使用所谓的v2模式。对于支持CLR 4的Visual Studio 2010这样的调试器来说，它支持两种调试模式，新的模式称为v4模式，在开始调试时可以选择，如图8所示。

图8　选择调试方式

对于.NET程序中的问题，大多数问题比较简单，用Visual Studio可以很容易解决。但也有复杂的问题，调试起来甚至比本地问题还要难：一个原因是因为隔了一层托管世界的“虚拟机”；另一个原因就是调试设施“先天不足”。很早就听说要改进，CLR4终于走出了关键一步，回归到了本地调试使用多年的成熟方法。当初为什么没这么做呢？这个问题留给大家讨论吧。

■ 责任编辑：董世晓（dongsx@csdn.net）

本期问题：

上期答案是：因为函数返回地址021b23e70对应的是托管方法中的代码，是以即时编译方式动态产生的，它在内存中的位置不属于任何模块的范围，因此调试器无法使用对待本地模块的方法找到它所对应的模块名。

本期的问题是：为什么重构后的新调试方式(v4模式)就不再与内核调试冲突了？

编者说明：

- 投稿信箱：contest@csdn.net
- 联系方式写在一个单独的TXT文件里，包括以下几项：
  1）姓名
  2）工作单位或学校
  3）电话联系方式
  4）邮寄地址
  5）E-mail地址
- 解答提交时间，最好早于当月15日。

下一代大规模增量索引平台——Percolator

■ 文 / 刘景龙

本文解读了Google近期公布的论文*Large-scale Incremental Processing Using Distributed Transactions and Notifications*，介绍了Percolator这一新的大规模增量索引平台。

继Google的三大基石GFS、MapReduce、BigTable之后，Google在2010年10月举行的OSDI（Operating Systems Design and Implementation）会议上公布了论文*Large-scale Incremental Processing Using Distributed Transactions and Notifications*，介绍了它最新的内容索引技术——Percolator。这项技术是Google下一代内容索引系统Caffeine的核心，其框架在抓取网页的同时进行文档处理，将平均延迟降低为原来的1%、平均文档寿命（document age）降低50%。

传统的索引系统通常采用多级索引的方式，按照网页的重要性将网页进行分级，分别按照小时、天、周对网页库进行更新。其特点是将网页收录到各级网页库时需要对全库进行处理。这种模式最大的缺点是新产生的网页或者信息不能被及时地收录到网页库中，而且定期做全库扫描也会造成计算资源的浪费。出现这个问题的根源在于现有MapReduce框架对于网页库的处理粒度过粗，一次全库更新需要几天时间才能完成。正因如此，Percolator细化了更新粒度，提供了对文档的随机访问，实现了对单个文档的处理，避免了MapReduce对全库的遍历。下面我来介绍一下Percolator的设计及实现。

设计

在最初的Percolator设计中，Google曾经考虑直接使用BigTable存储网页库。但对于构建网页库而言，需要比BigTable行原子性更强的一致性语义。而BigTable的可扩展性、容错以及负载均衡等设计是非常优秀的。为了避免重新发明“轮子”，Percolator在BigTable的基础上，通过两阶段提交实现了跨行、跨表事务以及Notifiication框架。

Percolator提供了对PB级网页库的随机访问功能，因此可以实现单独地处理每一个页面，从而避免使用MapReduce框架重建索引库时对全库的Scan操作。此外为了实现高吞吐以及高并发下的同步，Percolator支持ACID兼容的事务语义。

对增量索引系统而言，除了用户触发的操作外，很多处理流程是数据触发的。Percolator会根据数据的变化，触发后续的一系列操作（比如页面解析、内容抽取等）。为了满足此需求，Percolator提供了Notification语义。Notification语义类似于DBMS中的触发器，当Percolator中的某个cell类型数据发生变化时，就触发应用开发者指定的Observer程序。一系列Observer程序通过“责任链”的方式级联，并完成各自逻辑。

作为一个基础架构，Percolator继承了BigTable的一致性和容错模型，并且可以通过增加机器实现集群的线性扩展。此外，它提供了友好的开发接口，使索引系统的开发者专注于页面解析、抽取、分词等算法的开发，而不必被分布式系统中常见的一致性、容错等问题困扰。

如图1所示，Percolcator是构建在GFS和BigTable之上的，主要包含3个组件：Timestamp server、Percolcator worker和Chubby。

Timestamp server提供了统一时间的服务，它保证每次获取的时间戳单调递增。Timestamp server会持久化时间戳，以保证服务重启时时间戳的顺序性。

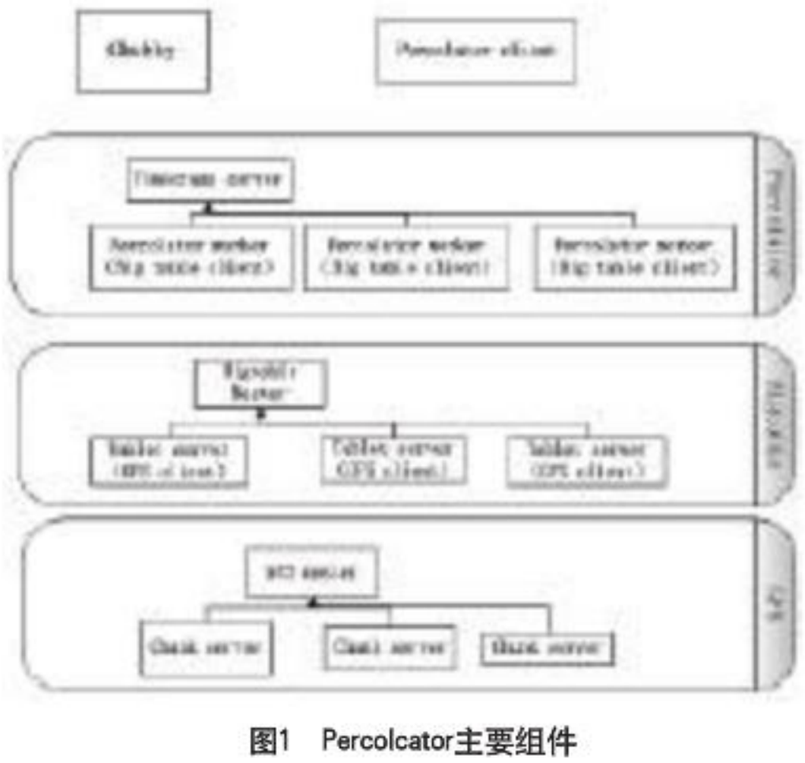


图1 Percolcator主要组件

Chubby服务提供了分布式锁服务，保证Percolcator在处理全局临界资源时可以互斥访问。

Percolcator worker是Percolator中的主要部分，实现了跨行、跨表事务和Notification机制。为了获得更好的吞吐量，Percolcator worker启动了多个BigTable client对BigTable进行并发访问。Percolcator worker并不持久化任何数据，换言之每个Percolcator worker都是无状态的，如果Percolator worker节点失效，Percolator client可以通过重试切换到另一个Percolator worker，从而不影响服务。

Percolator的设计主要是针对大规模数据存储，以及对网页索引具有实时性更新需求的应用。相对于传统的OLTP，Percolator没有集中的事务管理机制，如果有机器在进行事务的过程中失效，失效事务中的锁释放是一个比较大的挑战。由于Percolator是用于网页库建索引这样的线下服务，所以放松了对实时性的要求，采用一种延迟的方式清理锁。

实现

Percolator提供了类似BigTable的用户接口，Percolator中的cell通过BigTable中的五列来表示。其中lock、write、data 三列用于实现Percolator事务的功能。Notify和ack_O这两列是为了实现Percolator的notification机制。表1为Percolator在BigTable中的Schema。

接下来，我们通过Percolator中的两个主要的特性——事务和Notification框架，来介绍Percolator的实现机制。

表1 Percolator在BigTable中的Schema

Column	Use
c:lock	包含未提交事务的primary lock
c:write	事务提交的时间戳
c:data	存储数据
c:notify	observer可能需要被触发
c:ack_O	Observer “O” 已经被触发，保存其最后启动的时间戳

事务

Percolaor在BigTable的基础上，提供了跨行、跨表事务的ACID语义。Percolator中的事务是通过传统的两阶段提交实现。

下面举例说明Percolator是如何实现事务的。该实例实现了Bob账户转$7到Joe账户。

- 初始状态，Bob和Joe账户分别有$10、$2；write列中的6:data @ 5表示当前数据写入的时间戳是5。

key	bal:data	bal:lock	bal:write
Bob	6: 5: $10	6: 5:	6: data @ 5 5:
Joe	6: 5: $2	6: 5:	6: data @ 5 5:

- 开始事务，在时间点7对Bob的账户加Primary lock，并向Bob账户写入转账结果$3。

Bob	7:$3 6: 5: $10	7: I am primary 6: 5:	7: 6: data @ 5 5:
Joe	6: 5: $2	6: 5:	6: data @ 5 5:

- 在时间点7对Joe的账户加secondary lock，并且向Joe账户写入转账结果$9（secondary lock列中包含对primary lock的引用，row key为Bob）。

Bob	7: $3 6: 5: $10	7: I am primary 6: 5:	7: 6: data @ 5 5:
Joe	7: $9 6: 5: $2	7: primary @ Bob.bal 6: 5:	7: 6: data @ 5 5:

- 事务提交的第一阶段，提交primary，移除lock列的内容，在write列写入事务commit点的时间戳。

Bob	8: 7: $3 6: 5: $10	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:
Joe	7: $9 6: 5: $2	7: primary @ Bob.bal 6: 5:	7: 6:data @ 5 5:

- 事务提交的第二阶段，在Joe行中删除secondary lock，并且在Joe行写入commit点的时间戳。

Bob	8: 7: $3 6: 5: $10	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:
Joe	8: 7: $9 6: 5:$2	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:

Percolator就是通过以上五步完成事务的两阶段提交。对于分布式事务来讲，由于没有集中事务管理机制，其较大的困难就是在处理事务的过程中，Percolator client如果出现异常crash，如何清除已有的锁并重新使用被加锁的行。

Percolator采用的是比较消极的锁释放机制。如果事务A在执行过程中，Percolator client crash掉了，Percolator并不会主动释放事务A所占有的锁。如果事务B在执行过程中，使用到了事务A占有的数据，则由事务B负责锁的释放。

此时，就出现了一件比较棘手的事情：应当如何区分事务A是由于Percolator client crash导致不能完成，还是由于事务A长时间处于commit的状态？为了避免此类竞态条件，我们需要一种锁机制来同步应该执行事务B的cleanup任务或者事务A的commit任务。由于BigTable能够保证行的原子性，Percolator很自然地使用BigTable的一个cell作为锁（称为primary lock）：先抢到primary lock的可以执行，否则退出。

Notification机制

粗略地讲，Notification机制类似于DBMS中的触发器，即用户对cell的修改都会触发观察该cell的程序（我们把这个程序称为“Observer”）。多个Observer组成一个“责任链”，当被观察的cell被修改时，Observer链上的全部Observer将被触发。

在实现上，为了能够及时发现被修改的cell，Percolator在每一行中增加了一个BigTable列notify，用于标识被修改且没有触发执行Observer的cell。为了发现被标识为notify的cell，每个Percolator worker都会随机选取tablet，并进行Scan。对于被标识为notify的dirty cell，则触发Observer链。为了避免多个Percolator worker同时Scan相同的tablet，Percolator worker使用分布式锁服务（Chubby），在开始Scan一个tablet时对tablet进行加锁。对于网页库而言，这种周期性的对全表上PB级数据进行Scan是一个非常低效的操作，为了减少每次Scan的数据量，Percolator利用了BigTable按列存储的优势，将notify列指定为单独的locality group，保证Percolator worker只对notify列进行Scan，避免了全表扫描造成的对多余数据的加载。

此外，为了区分已被触发的Observer，Percolator增加了列ack_O（表示名字为O的Observer对应的ack），该列的内容为对应Observer的最后启动时间。当被观察的列被修改时，Percolator启动一个事务来处理Notification。该事务读取被观察的列以及其相关联的ack列，如果该column最后写时间大于ack_O中的时间戳，则认为该column对应的Observer需要被触发；否则，认为Observer已经被触发。

在Notification机制的实现上需要考虑两个重要的问题：如何防止多个Notification事务的无限循环？如果多个Notification事务触发了对同一个cell的修改，如何避免并发修改造成的正确性问题？

对于第一个问题，当前Percolator实现中并未对Notification事务的无限循环进行检测和防止，所以应用开发者必须在Observer开发中警惕此种情况，避免Notification无限循环的出现。

对于第二个问题，Percolator的解决方法是使每一个被触发的Observer为一个事务，如果启动了多个Observer同时修改一个cell，则事务锁机制可以避免重复修改引入的正确性问题。

总结

本文简单介绍了下一代大规模增量索引平台Percolator所要解决的问题以及实现的方法。该平台构建在Big Table和GFS上，利用了GFS的数据安全以及BigTable的行原子性、负载均衡以及服务容错等优越特性，提供了稳定、可扩展的增量索引构建平台，并且相对于传统索引技术而言，带来了搜索结果的实时性和集群资源利用率的全面提升。在一窥Google基础架构的同时，我们可喜地看到，作为Percolator中最主要的两大功能的身影已经出现在HBase社区的开发计划中（Transaction和Coprocessor），下一代大规模增量索引平台离我们并不遥远。

作者简介 | 刘景龙

百度基础架构部工程师。2010年加入百度，曾就职于阿里巴巴云计算公司，主要从事分布式文件系统和分布式表格系统的设计与开发工作。对Hadoop及HBase有较深入的研究。

■ 责任编辑：董世晓（dongsx@csdn.net）

我的工具箱

■ 文 / 周爱民

架构师的工作既要考虑整体、又要顾及细节，他们是如何做到这一点的呢？本文作者使用软件工具的心得一定会给你带来启发。

我习惯的桌面环境是Windows。在开始一些具体工具的推荐之前，首先建议你选择一个能在创建下载任务的同时就可以输入备注和选择分类的下载软件。正是这样的功能使我得以建立并方便地维护着一个几十GB的软件目录。我习惯在下载工具的备注中搜索，以找到某些知道功能、但不知道名字的工具。

正因为我喜欢用工具来提高做事效率，所以我的机器上安装了千余个大软件或小工具，近十年的时间里我一直保持着相同的系统环境、目录结构和工具清单。管理它们有两种方法，第一种是合理的程序分组与使用桌面工具栏，第二种方法则是利用“Alt +R”热键。一般来说，在安装一个软件后我只会留下主程序的快捷方式，然后分类放在一些文件夹中。最终我会把（这些仅仅包含快捷方式的）文件夹直接拖到Windows任务栏的“开始”按钮上面，以得到跟“程序”并列的程序组，如图1。

图1　仅仅包含快捷方式的文件夹组成了程序组

此外可以创建类似IE Quick Launch的工具栏（我称为qLink，在任务栏上点鼠标右键，选工具栏→新建工具栏即可），它们的分工是：IE Quick Launch是最常用的长时工作的工具，例如IDE、Word等；qLink中是临时频繁使用的小工具或帮助文档的快捷方式，例如Win32 Spy、语言手册等。这些桌面工具栏中也可以创建子文件夹，以形成分层的菜单。通过这些方法，我的操作系统中共安装了300多个不同的软件，却不显混乱。

其他的工具大致都是绿色的（免安装）——通常是单个可执行程序，或者创建了一个快捷方式并赋以一个简洁的英文名字。这些可执行程序或快捷方式被放在同一个目录中（例如C:\Tools），最后我将它们的路径添加到PATH环境变量里。这样一来，我可以按下“Win + R”键，直接输入可执行程序或快捷方式的名称来启动它。这一类软件约有600多个。

先说两个给我工作带来最大收益的工具。其一是ManicTime，它是一个免费的、功能强大的时间跟踪软件。与多数类似软件不同，ManicTime是绝对的桌面软件，不用担心联网泄露你的个人信息。ManicTime总是运行在后台，多数情况下你什么也不用管，如果你为这个软件花费太多时间，那就失去了它存在的意义了。但如果你有一些特殊时间在做一些特殊的事情，ManicTime无法自动记录的话，你可以为此打个标签（Tag），这在你以后的分析中会相当 有用。方法是“在Tag时间条上拖选一个

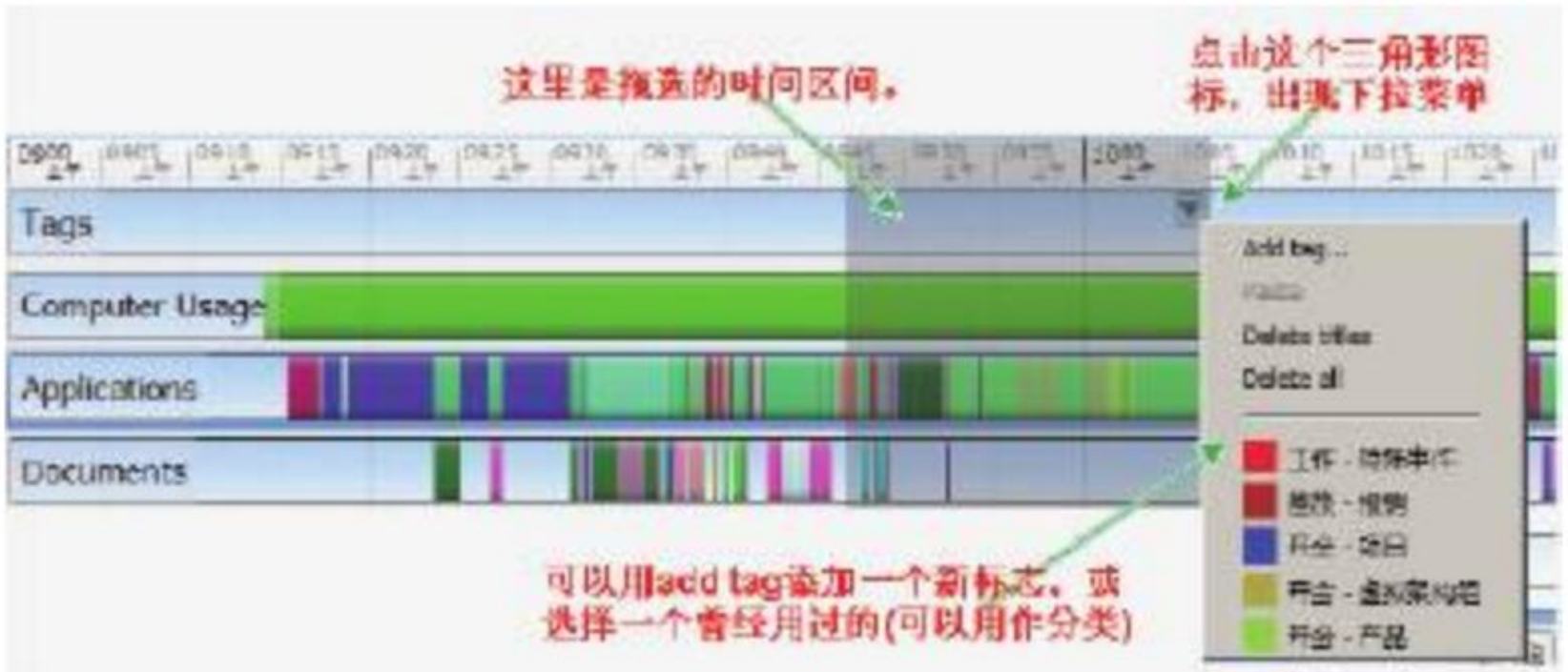


图2 ManicTime为优化时间管理提供了可靠依据

区间，然后点击菜单”，如图2所示。

ManicTime的“统计”中缺省有许多分析模板。首先，你需要知道一个要点：至少在后台运行ManicTime几小时或半天时间，才能开始使用“统计”功能，否则将没有任何可用的数据项。接下来最重要的是统计中的自定义（Custom）分析模板，免费版本中你只能有一个这种模板，但可以添加任意多个分析项。例如我需要分析我的工作时间，那么使用“添加”，在时间轴（Timeline）中选应用程序（Application）。然后ManicTime会列出你所有用过的软件。我会选中以下软件：

- Eclipse
- Mindmanager
- Editplus
- Beyond Compare

…

基本上，在使用这些软件时，我都是在做“开发工作”。当然，你也可以把Office Word之类的选进去，这取决于你的工作方式或内容。通常情况下，我会把整个的计算机使用时间也统计进去，作为参考线。同样的方法，添加一个时间轴，选择计算机应用（Computer Usage）中的活动（Active）。这样我们就创建了两个分析项，它会显示在Custom分类统计中，如图3所示。

ManicTime还会自动记录并排序你经常去的网站/网页，或打开的文档。一段时间之后，你可以对它们进行归类，例如什么时间在做程序或查资料、什么时间看新闻或娱乐、什么时

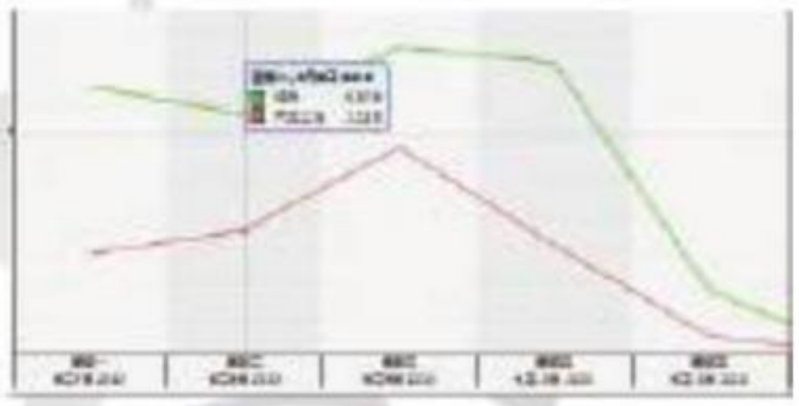


图3 ManicTime中自定义分类统计

间在泡论坛或者写博客等。所以ManicTime的使用时间越长，能对你的工作分析的准确性就越高。

利用前面提到的“添加标签（Tag）”的方法，我们可以在时间轴上标记一些特殊事务，例如会议。这样就可知道我们为哪些事投入了多少时间成本。

通过这个软件，在一段时间之后，我们对自己的认识将会很有不同。至少可以开始了解自己为什么会停下来不想工作，或者工作总没有效率，或者看到是哪些事务在零敲碎打地夺走我们的时间。所以，建议你现在就开始使用ManicTime，真的。

在架构工作中对我有很大影响的工具是MindManager，不过介于网上介绍这个软件的文章已经填坑满谷，所以我换个小众一点的东西来介绍。这是在一个名为SimpleGraph的开源组件中提供的DMEO程序，它可以简单地做出图4这样的图形来（类似的工具，我还推荐来自http://logicnet.dk/的Diagram Designer）。

你可以随意组装一些资料，并用你希望的形式画这种既不是流程图也不是UI样稿的

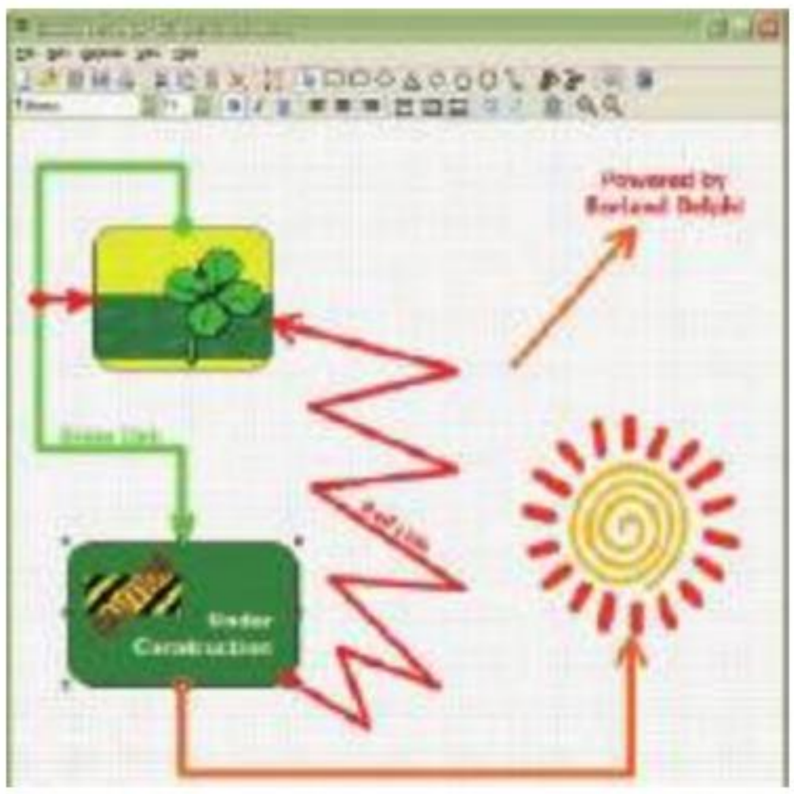


图4　简单的思维导图对设计架构很有帮助

图形，从而来记录你的思考过程。我之所以在架构工作中用到它，就是因为它不用某种特定的架构方法来约束我的思想，又有足够的架构和思维表达能力。相反的，如果你的架构成型了，只是需要表达出来，那么你可以选择EA（Enterprise Architect）这类标准的工具。

我的工作有20%以上是在DOS中完成的，有大量开发工作时，这个比例可能会上升到50%~70%。因此一些精品的DOS工具是必备的，例如grep。如果你仍然不会用grep——哪怕是GUI版本——那么你绝对算不上一个好的程序员。另外我推荐listdlls、junction和vsubst，你建议你立即去网上找到它们，然后把它们用起来，这是一些简单而又完美的工具。

我也经常会写一些批处理或JScript脚本来配合工作，我很推荐使用JScript来写一些小工具。如果你希望最终界面表现得好一些，那么你还可以将一个.HTML文件重命名为.HTA，再加上一点点改写，就可以变成一个类似.exe那样可执行的桌面工具了——这一切的秘密在于Windows已经缺省安装了一个名为MSHTA.EXE的程序，它会把.HTA当成一个应用（Application）来打开。

我习惯用大量快捷键来组织我的工作，不过仅实用性而言，我只推荐用HoeKey来定义和管理这些快捷键。而且在所有HoeKey支持的快捷键中，我也只推荐两个。其一是最简单的：在“当前位置”打开DOS窗口。这只需要在HoeKey中添加一行配置信息：

```
@_d=Run|cmd.exe|||%c
```

其效果是在你按下“alt + shift + D”键时就会打开一个DOS窗口，目录设在你的“当前位置”。这个当前位置是你在资源管理器中的当前目录，或你打开某个文件对话框里所在的目录，或者任何可以判断出“这是个目录”的地方。这对于常常需要在DOS和Windows桌面中切换的开发人员来说，实在是太方便了——例如你在Eclipse开发中需要进入DOS去用Git来签出当前代码等。

另外一个快捷键则取决于你的习惯，你可以任意定义一个“Win + ?”键——最好是左手一次就可以全部按到的组合键，将这个键关联到一个.txt或其他什么类型的便签工具。好了，你就可以有一个随手记的功能了。我常常用这种方法来记录一点突发奇想的东西，好处在于：打开一个.txt的速度可能是整个Windows操作中最快的，并且我根本不需要任何格式或者排版的支持。我关注的只有一件事：快点让我记下来，别让这个念头溜掉。这也只需要添加一行配置信息：

```
~t=Run|" D:\My Documents\qnote.txt"
```

HoeKey有非常完善的在线文档，因此这里就不介绍如何将上面这些配置用在HoeKey中了。

编程方面我最常用的是Editplus（我已经很久很久只写JavaScript代码了）以及记事本（Notepad.exe）。代码比较则一直用Beyond Compare。

最后，除了上述这些软件与程序之外，我唯一要强调的事情是：用好Office，尤其是Excel，会给你的工作带来莫大的收益。当然，即使是在Windows环境中，你仍然可以选择OpenOffice。

作者简介 | 周爱民

国内软件开发界资深软件工程师，技术作家。有10余年的软件开发、项目管理、团队建设的经验，现任支付宝公司业务架构师。

■ 责任编辑：高松（gaosong@csdn.net）

微软Dryad分布式并行计算平台

■ 文 / 高阳

微软于2010年12月21日发布了分布式并行计算基础平台——Dryad测试版，成为Google MapReduce分布式数据计算平台的竞争对手。开发人员能够借此在Windows或者.NET平台上编写大规模的并行应用程序模型，单机上编写的程序可以轻易运行在分布式并行计算平台上，并可利用服务器集群对数据进行并行处理，程序开发人员操作数千台机器时，无需/须关心分布式并行处理系统方面的细节。本文将重点讲述Dryad平台的功能原理及应用。

Dryad概述

Dryad和DryadLINQ是微软硅谷研究院创建的研究项目，主要提供一个分布式并行计算平台，DryadLINQ提供一种高级语言接口，使普通程序员可以轻易进行大规模的分布式计算，它结合了微软Dryad和LINQ两种关键技术，被用于在该平台上构建应用。Dryad与微软体系结构中的位置关系，如图1所示。

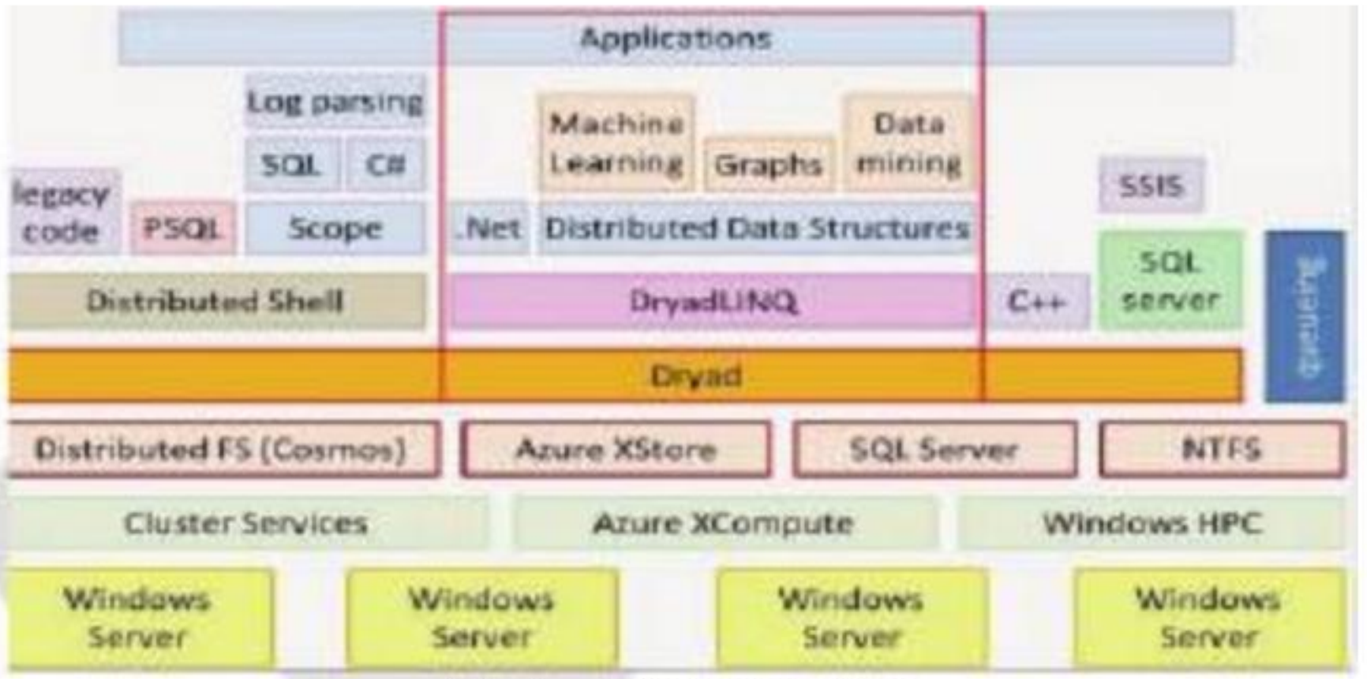


图1 Dryad与微软体系结构的关系

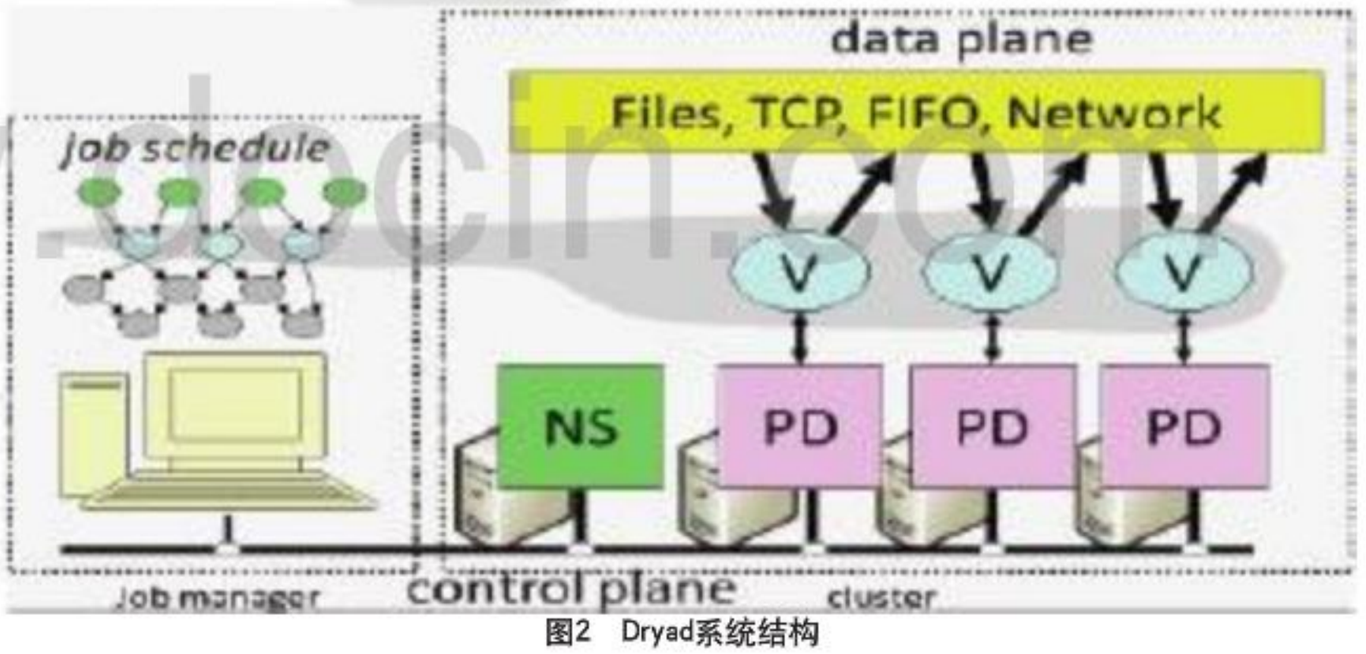


图2 Dryad系统结构

Dryad同MapReduce一样，不仅是一种编程模型，也是一种高效的任务调度模型，这种编程模型并不仅适用于云计算，在多核和多处理器以及异构机群上同样有良好的性能。

Visual Studio 2010 C++有一套并行计算编程框架，支持常用的协同任务调度和硬件资源管理，通过Work stealing算法以充分利用细颗粒度并行的优势，来保证空闲的线程依照一定的策略建模，从所有线程队列中“偷取”任务执行，所以能够让任务和数据粒度并行。

Dryad与上述并行框架相似，不同的是Dryad被设计为伸缩于各种规模的集群计算平台，无论是单台多核计算机还是由多台计算机组成的集群，甚至拥有数千台计算机的数据中心，都可以从任务队列中创建的策略建模来实现分布式并行计算的编程框架。

Dryad系统架构

Dryad的总体构建支持有向无环图（Directed Acycline Graph，DAG）类型数据流的并行程序。整体框架根据程序的要求完成调度工作，自动完成任务在各个节点上的运行。在Dryad平台上，每个工作或并行计算过程被表示为一个有向无环图。图中的每个节点表示一个要执行的程序，节点之间的边表示数据通道中数据的传输方式，其可能是文件、TCP Pipe、共享内存等，为了支持数据类型需要针对每个类型有序列化代码。

如图2所示，当用户使用Dryad平

台时，首先是需要在任务管理器节点上建立自己的任务。每一个任务由一些处理过程以及在这些处理过程数据传递组成。任务管理器获取无环图之后，便会在程序的输入通道准备，当有可用机器的时候便对它进行调度。随后JM从命名服务器那里获得一个可用的计算机列表，并通过一个维护进程（PD）来调度这个程序。

系统组件：

- 任务管理器：每个Job的执行被一个Job Manager控制，该组件负责实例化这个Job的工作图；在计算机群上调度节点的执行；监控各个节点的执行情况并收集一些信息；通过重新执行来提供容错；根据用户配置的策略动态地调整工作图；
- 计算机群：用于执行工作图中的节点；
- 命名服务器：负责维护Cluster中各个机器的信息；
- 维护进程：进程监管与调度。

从总体来看，传统的Linux/Unix管道是一维管道，每个节点在管道中是单个的程序。而Dryad的执行过程就可以看做是一个二维的管道流的处理过程。其中，每个节点可以具有多个程序的执行，通过这种算法可以同时处理大规模数据。

如图3所示，在每个节点进程上都有一个处理程序在运行，并且通过数据管道的方式在它们之间传送数据。二维的Dryad管道模型定义了一系列的操作，用来动态建立并且改变这个有向无环图。这些操作包括建立新节点，在节点间加入边，合并两个图以及对任务的输入和输出进行处理等。

Dryad模型算法应用

DryadLINQ可以根据程序员给出的LINQ查询生成可以在Dryad引擎上执行的分布式策略算法建模（运算规则），并负责任务的自动并行处理及数据传递时所需要的序列化等操作。此外，它还提供了一系列易于使用的高级特性，如强类型数据，Visual Studio集成调试，以及丰富的任务优化策略（规则）算法等。这种模型策略开发框架也比较适合采用领域驱动开发设计来构建“云”平台应用，并能够较容易做到自动化分布式计算。

通过并行算法策略建模，可以有效的控制数据的颗粒度，当程序运行在Dryad分布式并行平台时候，可最大化的提高分布式并行运算效率。

我们经常会遇到，所开发的网站/系统无法承载大规模用户并发访问的问题。解决该问题的传统方法是通过数据库提供的访问操作接口来保证处理复杂的查询能力。当访问量增大，单数据库处理不过来时便增加数据库服务器的数量。

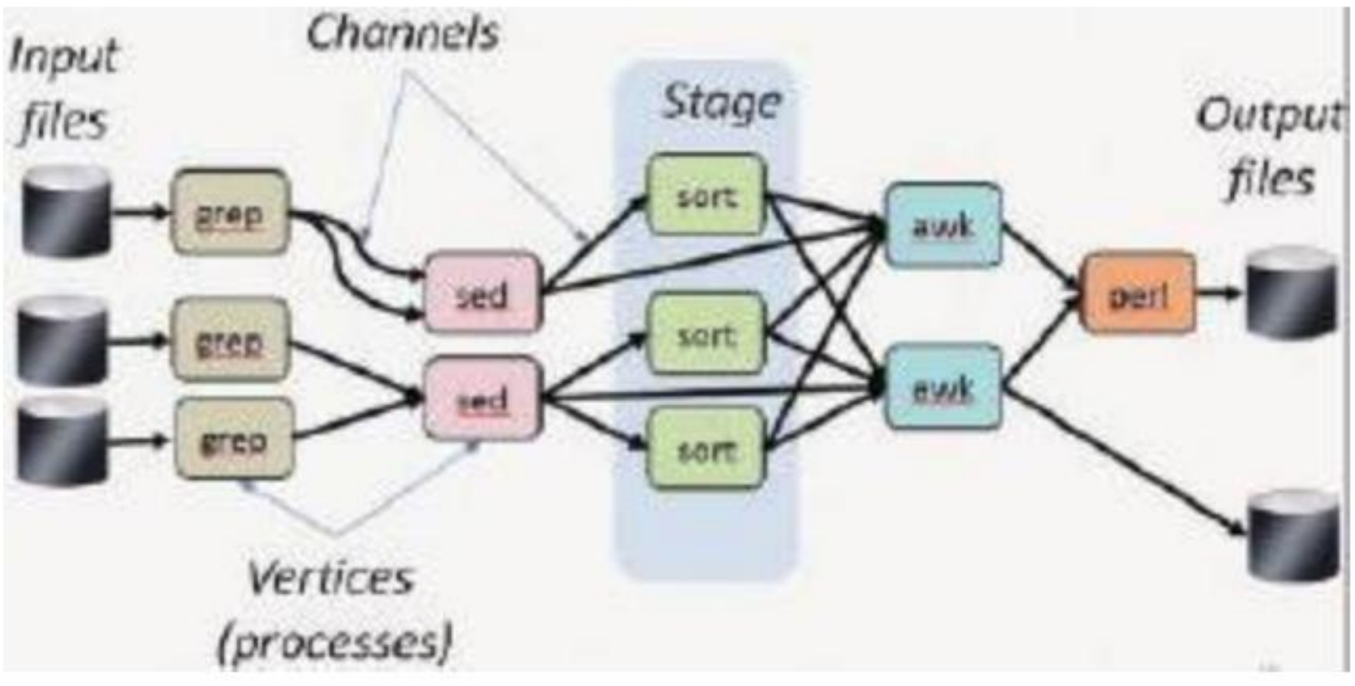


图3 Dryad任务结构

但如果增加了3台服务器，再把用户分成三类：A（学生），B（老师），C（程序员）。每次访问的时候，会先查看用户属于哪一类，然后直接访问存储那类用户数据的数据库，这时处理能力增加了三倍，而我们已经实现了一个分布式的存储引擎过程。

如果这3台服务器也承载不了更大的数据要求，需要增加到5台服务器，那必须更改分类方法把用户分成五类，然后重新迁移已经存在的数据，这时就需要非常大的迁移工作，这种方法显然不可取。另外，当群集服务器进行分布式计算运行的时，每个资源节点处理能力可能有所不同（如不同硬件配置的服务器等），如果只是简单地把机器直接分布上去，性能高的机器得不到充分利用，性能低的机器也处理不过来。

Dryad解决此问题的方法是采用虚节点。把上面的A B C三类用户都想象成一个逻辑上的节点。一台真实的物理节点可能会包含一个或者几个虚节点(逻辑节点)，视机器的性能而定。我们可以把那任务程序分成Q等份(每一个等份就是一个虚节点)，这个Q要远大于我们的资源数。现在假设我们有S个资源，那么每个资源就承担Q/S个等份。 当一个资源节点离开系统的时候，它所负责的等份要重新均分到其他资源节点上，一个新节点加入的时候，要从其他的节点“偷取”到一定数额的等份。

在这种策略建模算法下，当一个节点离开系统的时候，虽然要影响很多节点，但是迁移的数据总量只是离开那个节点的数据量。同样，一个新节点的加入，迁移的数据总量也只是一个新节点的数据量。之所以有这个效果是因为Q的存在，使得增加和减少机器的时候不需要对已有的数据做重新哈希（D）。这个策略的要求是Q>>S（存储备份上，假设每个数据存

储N个备份则要满足Q>>S*N）。如果业务快速发展，使得不断增加主机，从而导致Q不再满足Q>>S，那么这个策略将重新变化。

通过上述的论述，我们可以看到Dryad通过一个有向无环图的策略建模算法，提供给用户一个比较清晰的编程框架。在此编程框架下，用户需要将自己的应用程序表达为有向无环图的形式，节点程序则编写为串行程序的形式，然后用Dryad方法将程序组织起来。用户不需要考虑分布式系统中关于节点的选择，节点与通信的出错处理手段都简单明确，内建在Dryad框架内部，满足了分布式程序的可扩展性、可靠性和性能的要求。

使用DryadLINQ

使用DryadLINQ编程，普通的程序员编写大型数据并行程序能够轻易运行在大型计算机集群里。DryadLINQ开发的程序是一组顺序的LINQ代码，它们可以针对数据集做任何无副作用的操作，编译器会自动将其中数据并行的部分翻译成并行执行的计划，并交由底层的Dryad平台完成计算，从而生成每个节点要执行的代码和静态数据，并为所需要传输的数据类型生成序列化代码。

DryadLINQ使用和LINQ相同的编程模型，并扩展了少量操作符和数据类型以适用于数据并行的分布式计算。并从两方面扩展了以前的计算模型（SQL、MapReduce、Dryad等）：它是基于.NET强类型对象的、表达力更强的数据模型和支持通用的命令式和声明式编程（混合编程），从而延续了LINQ“代码即数据”的特性。

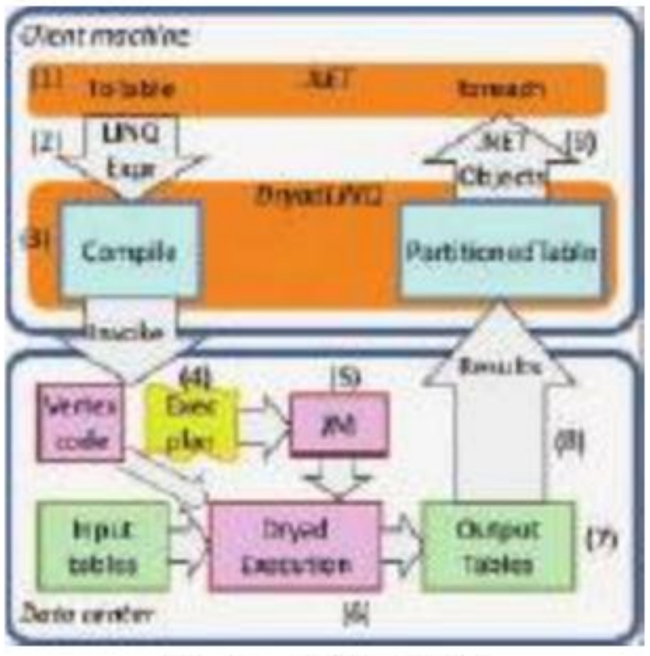


图4 DryadLINQ系统架构

如图4所示，LINQ本身是.NET引入的一组编程结构，它用于像操作数据库中的表一样来操作内存中的数据集合。DryadLINQ提供的是一种通用的开发/运行支持，不包含任何与实际业务、算法相关的逻辑。DryadLINQ使用动态的代码生成器，将DryadLINQ表达式编译成.NET字节码。这些编译后的字节码会根据调度执行的需要，被传输到执行的机器上去。字节码中包含两类代码：完成某个子表达式计算的代码和完成输入输出序列化的代码。

DryadLINQ表达式代码示例片段如下：

```
Collection<T> collection;
bool IsLegal(Key k);
string Hash(Key);
var results = from c in
collection where IsLegal(c.key)
select new { Hash(c.key),
c.value};
```

这种表达式并不会被立刻计算，而是等到需要其结果的时候才进行计算。DryadLINQ设计的核心是在分布式执行层采用了一种完全函数式的、声明式的表述，用于表达数据并行计算中的计算。这种设计使得我们可以对计算进行复杂的重写和优化，类似于传统的并行数据库，从而解决了传统分布式数据库SQL语句功能受限与类型系统受限问题，以及MapReduce模型中的计算模型受限和没有系统级的自动优化等问题。

另外，在MapReduce编程方式下，应用程序编写人员需要关注与自己的应用逻辑如何使用Map函数以及Reduce函数进行表达。在Dryad编程模式中，应用程序的大规模数据处理被分解为多个步骤，并构成有向无环图形式的任务组织，由执行引擎去执行。这两种模式都提供了简单明了的编程方式，使得应用程序开发人员能够很好的驾驭云计算处理平台，对大规模数据进行处理。Dryad的编程方式可适应的应用也更加广泛，通过DryadLinq所提供的高级语言接口，使应用程序员可以快速进行大规模的分布式计算应用程序的编写。

写在最后

通过以上介绍，我们可以体会到Dryad的诸多优点，如DryadLINQ具有声明式编程并将操作的对象封装为.NET类方便数据操作、自动并行化、Visual Studio IDE和.Net类库集成、自动序列化和任务图的优化（静态和动态（主要通过Dryad API实现））、对Join进行了优化，得到了比BigTable+MapReduee更快的Join速率和更易用的数据操作方式等。不过，Dryad和DryadLINQ也同样具有局限性。它更适用于批处理任务，而不适用于需要快速响应的任务；这个数据模型更适用于处理流式访问，而不是随机访问。虽然目前Dryad还是测试阶段尚未大规模普及，但是微软已经在AdCenter的生产系统中使用Dryad。

与MapReduce不同，DryadLINQ使用的是.NET的LINQ查询语言模型，并且Dryad是针对运行Windows HPC Server的计算机集群设计，而非兼顾Linux，而目前Apache的Hadoop环境只支持Linux。目前而言，高性能计算市场被Linux所占领，但是笔者相信Dryad平台在将来一定具有很广泛的发展前景，尤其对.NET开发人员来说也是一次很重要的技术革新机遇。

作者简介 | 高阳

微软ASP.NET MVP，10多年的管理与程序设计经验，7年左右的.NET应用程序开发经验，对于微软.NET企业应用开发与架构设计有较为深入的理论基础与实践经验。

■ 责任编辑：高松（gaosong@csdn.net）

模块化的JavaScript库：Tangram

■ 文 / 雷志兴

Tangram是一个适用面广、高度模块化的JavaScript库。内含大量基础方法和Javascript组件，开发人员能基于它便捷地完成开发工作。本文将介绍Tangram的历史、设计原理和系统架构，旨在帮助各位读者更好地了解Tangram。

历史

要理解Tangram的使用场景和设计思想，需要从它的历史渊源说起，其实这也是百度Web前端库的发展史。

2005–2007年，百度前端处于洪荒年代，没有JavaScript库，仅在部门Wiki中存在一些函数积累，工程师开发时，Copy+Paste以后稍作修改便成为产品的一部分。

2007年起，部分产品逐渐开始基于开源库（Prototype、YUI等）进行修改，形成产品内部的JS库，积累也逐渐增多。但是开源库的功能相对百度的需求都过于复杂、体积庞大，部门内出现需要百度JavaScript库的呼声。

2008年初，一些有想法的工程师利用项目闲暇时间，整理编写第一个通用JavaScript库：FE。FE库在百度社区类产品线得到了较广泛的使用，从那时起逐渐形成了模块化、无侵入的设计思想。

2009年6月，由于FE库没有固定的维护人员，同时在接口组织、可维护性上还有很大的提升空间，经过讨论，决定基于公司内多个JavaScript库，由专人负责，开发出一个新的类库。于是Tangram这个全新的基础库诞生了，经过一年多时间发展，Tangram已经包含开发中大多数常用的功能与组件，成为了百度应用最广泛的JavaScript库。

2010年12月底，Tangram正式开源，现已正式发布两部分。一部分是基础库，包含DOM操作、Ajax等基础功能；另一部分是组件库，包含大量可重用的UI控件以及动画效果等。除此之外，我们还基于Tangram开发了在线编辑器、无线库等子项目。

设计原理

七巧板——Tangram的中文名，这是一个古老、简单的模块化系统。Tangram库正如七巧板一样，它的核心设计思想就是模块化。

模块粒度小

在国内，由于互联网的连接速度仍然比较慢，同时用户的计算机与浏览器相对老旧，因此前端开发人员一直都很关注带宽占用和加载速度。市面上的JavaScript库，最小的模块一般是一组功能方法（如DOM操作的相关方法）或者一个组件（如对话框）。开发人员在选择时存在一个两难问题：如果库体积精简，二次开发量就大；如果库功能丰富，就会有冗余代码。

为解决这个问题，Tangram采用了更小的模块拆分。在Tangram基础库中，每一个函数方法是一个独立的模块；在UI组件中，最小的模块单位是组件插件。开发人员通过在线代码选择功能，获得定制后的Tangram放入自己的产品中。

因此，Tangram能够不断补充功能模块，而使用者始终能按照需求，获得体积最精简的代码，这在功能丰富与体积精简之间达到了平衡。同时，每一个模块都是独立而且简单的，开发人员也可以很方便地对其进行研究，了解原理进而更好地使用。

图1　Tangram的代码选择功能界面

易于封装、扩展

Tangram面向的产品跨度广，光是百度内部就有搜索产品、社区产品、商业产品三大系列。不同的产品对JavaScript库的要求也不尽相同：搜索产品要求体积精简，社区产品希望使用方便，商业产品看重功能齐全。

很显然，Tangram不可能直接满足所有产品的需求，只能在封装和扩展的便利性上下工夫。首先，基础库中大多是静态方法，对封装十分友好。而在组件库中，提供了组件插件、公用行为等机制，让使用者可以很方便地扩展出自己想要的功能组件。使用者很方便地就能基于Tangram，选择合适的模块，封装一套符合产品自身特点的JavaScript

库和一个个独立的功能组件。

当然，在Tangram内部，也会利用这个特性完成一些较高层次的封装。如将DOM和Event封装成Element，提供链式调用功能。

无侵入式设计

面对多年前的遗留系统，或者含有第三方JavaScript代码的系统时，总能看到页面上泛滥着各种全局变量，或者对Prototype的修改，甚至会存在同一个JavaScript库的不同版本。开发者常常会因为代码之间的相互冲突而烦恼不堪。

为了最大程度地避免冲突和污染，Tangram仅仅暴露一个变量到当前作用域，使用者完全可以把Tangram放在闭包中使用，这样即使页面上含有两套不同版本的Tangram，它们之间也不会有任何冲突。同样地，组件库也采用了无污染的设计，不会对页面上的现有DOM元素、业务代码产生影响。

系统架构

Tangram从一开始就致力实现一个结构清晰、层次分明的系统。系统架构如图2所示。

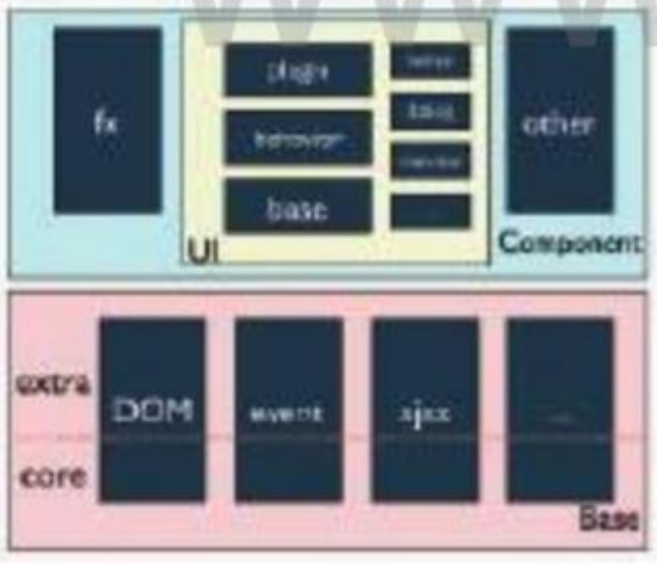


图2 Tangram系统架构图

基础库

Tangram底层是基础库，包括18个功能包，如DOM、Event、Ajax、Page等，满足开发中对基础方法的需求。基础库被分作两部分：core和extra。core是常用方法，gzip压缩后体积尚不足10KB；extra则是其余不够常用的方法。这是统计了用户对接口的依赖程度后进行的划分。为保证基础库方法的精简，所有基础库方法不做参数检查，由调用者或者二次封装自行保证。此外，整个Tangram库都有很强的一致性，同类的方法接口命名和参数形式都很一致。

组件库

组件库在基础库之上，其中又以UI组件结构较为复杂。所有的21个UI组件都是基于UI Base开发的，UI Base实现了UI组件的统一创建，插件与行为机制，代码也足够精简，gzip压缩后体积仅1KB。

UI组件的插件与行为

如前文所述，组件的功能丰富和体积精简之间存在矛盾。特别是在产品不需要组件的某些内置功能时，那部分代码就成了累赘。为了调和这种矛盾，Tangram在UI体系中采用了插件机制。每个UI组件都被拆分成了一个小巧的UI核心，与一组功能独立的插件。

UI核心仅完成数据结构定义，控件DOM结构和其他控件基本功能，所有非必需功能都由插件提供。插件可以在代码选择时，由使用者根据需要，自由选择。如对话框核心仅仅完成对话框的创建、开启、关闭与更新，由插件提供拖拽、缩放大小、键盘支持、模态窗体等附加功能。

此外，针对UI组件间的共同点，我们抽取出了UI behavior。UI行为能被自由添加到任何组件上，这个附加动作甚至可以来自用户调用，如将draggable行为附加到组件上以后，组件就拥有了拖拽行为。

开发、发布过程

Tangram现在由专门的团队负责开发和维护，为了保证库的持续发展，一套严格、可遵守的开发流程是关键。Tangram的开发分四个阶段。

调研、接口设计：调研使用场景与问题来源，同时参考同类组件，完成功能定义与接口设计，供小组评审。如果功能较复杂，需要提供一个完成基本功能的Demo，用于验证设计的正确性。

开发：完成代码，手动测试用例与基础自动用例，基础库模块采用测试驱动的方式进行开发。

代码评审：将代码提交到CodeReview平台上，全组参与评审，一个模块一般需经历2～3轮评审修改。评审完毕后，代码会被提交至GitHub的Dev分支上，由QA编写完整的自动化测试用例进行测试。

在开发过程中，我们尤其注重设计和代码评审阶段。一般来说，一个模块的设计时间要占到整个开发周期的50%以上，而代码评审时间在30%左右。值得一提的是，注释在Tangram中占有很重要的地位，所有API文档都是通过注释生成，因此在代码评审时，也会重点关注注释的准确性和完整度。

当Dev分支上的功能累积达到Roadmap目标，进入发布阶段。由发布负责人Review所有新增功能与Bug修复，整理ChangeLog，同时QA负责人确认自动化用例的覆盖率，进行完整回归。测试完毕后，会将相应代码合并到Master分支，打上版本Tag。

文档与源码

Tangram Base和Component都将在兔年春节前后发布一个新的稳定版，此外，Tangram编辑器也有计划于2011上半年开源，同时无线库也在紧张的开发中，敬请期待。

作者简介 | 雷志兴（aka. berg）

现就职于百度公司Web前端研发部，负责百度前端通用技术规划、解决方案等相关工作。

■ 责任编辑：高松（gaosong@csdn.net）

新产品&新工具

Doit.im

好的GTD工具可成倍提高工作效率。Doit.im是一款基于AIR的GTD工具。有苹果简洁风格的界面设计，Twitter式的收集箱，并且支持拖拽效果，非常容易上手。目前提供了各个平台的客户端，包括Windows、Mac、Android、iPhone、iPad等。

JsDoc Toolkit：JavaScript 文档利器

随着Web 2.0的风靡，JavaScript已成为一门被人们重新认识的编程语言，由于大量JS开源框架出现，越来越多JavaScript开发的问题暴露出来。其中，JavaScript文档维护也成为开发者亟待解决的一个难题。JsDoc Toolkit可以很好地解决这一问题。它是发布在Google Code上的一个开源项目，和其他语言的文档工具一样，可以自动从JavaScript代码中提取注释生成格式化文档。

财务软件GnuCash

GnuCash是一款GNU发布的适合个人或小型企业的财务软件。该软件允许用户跟踪银行账户、股票、收入和支出，具有日常纸质账簿类似的直观性。基于专业的会计理念可以确保平衡的账簿和精准的报表。

GnuCash新发布了2.4.0版本。主要特性包括：可以使用SQLite3的SQL数据库或者MySQL和PostgreSQL数据库存储数据；可以使用WebKit作为当前的GtkHTML HTML引擎，用来显示报告和图表等。

Mozilla的新编程语言

Mozilla正在开发一个新的编程语言，名为“Rust”，由Web语言领军人物Brendan Eich、Dave Herman以及Mozilla公司的Graydon Hoare合力开发。

开发这个新语言的目的是为了解决一个顽疾：软件的演进速度远低于硬件的演进，以及软件在语言级别上无法真正利用多核计算带来的性能提升。Rust是针对多核体系提出的语言，并且吸收一些其他动态语言的重要特性，比如不需要管理内存、不会出现Null指针等。

开源压力测试框架fuload

fuload是为解决大型服务的压力测试或性能测试而诞生的，可以通过fuload来对等待上线或者已经上线的服务进行压力测试，并能通过详细的报表得出对服务的客观评价。

操作系统Verve

微软近期发布一份白皮书，介绍了新的非Windows操作系统Verve。Verve当前还只是微软研究院开发的一个原型，作为操作系统和运行时系统，主要是为了确保类型和内存的安全性。Verve是从另一个非Windows系统Singularity中剥离出来的，Singularity平台项目则是一款用托管代码编写的以研究为目的的操作系统。

微软表示："汇编语言（TAL）和霍尔逻辑可以确保低级代码中不会出现多种错误，使用汇编语言和霍尔逻辑来实现一个新操作系统Verve可保证在安全性方面的高度自动化、静态验证。"

云系统Jolicloud

Jolicloud OS是一款基于Ubuntu、专为上网本定制的操作系统。与Chrome OS 类似，Jolicloud OS也定位于云系统，内置了大量的在线应用。不同的是，它可以像Ubuntu一样直接通过 Ubuntu官方源来安装本地软件。根据Jolicloud官方Blog的通告，最新的 1.1 版已正式发布。

phpMyFAQ /nginx 新版本

完全数据库驱动的FAQ系统phpMyFAQ 2.6.12发布。添加对MariaDB的支持、添加LDAP示例配置文件、更新翻译、修复错误。

高性能的HTTP和反向代理服务器nginx 0.9.3发布。修复了nginx不能搭建在Solaris上等问题。

微软正式发布Webmatrix建站工具

经过三个测试版本，微软正式发布了最终版本的WebMatrix Web开发平台。该产品首次推出是在2010年7月，目的是提供一种简单、一体化的建站方案。它提供了网站所需的所有工具：Web Server、数据库、Web框架和开发环境。同时与微软的 Visual Studio 和 SQL Server 产品保持兼容。

GParted

GParted是一款以直观的图形环境为硬盘划分分区的工具，与著名的Partition Magic表现同样出色。最新发布的GParted 0.7.1提高对主板BIOS RAID的支持，修复了dmraid分区路径名称错误、MBR中没有反映GPT的分区表等问题。

Python解释器PyPy提速

用Python语言实现的Python解释器PyPy 发布了1.4版。开发者表示，PyPy 1.4意义重大，因为它是首个翻译速度比CPython快的版本，他们正将PyPy作为日常开发工具使用。新版的主要功能包括：改进JIT编译器；完整兼容virtualenv；更快的正则表达式。

本地存储开发插件：Rookie

Rookie是一款用于Web开发实现本地存储功能的脚本小插件，采用JavaScript和SWF文件通信，通过调用Flash SharedObject对象完成本地数据存储。Rookie借助网上已有方案的思路进行了一些功能拓展与完善，更方便第三方使用。目前，已兼容主流浏览器；支持跨域读写本地存储；有安全可控、支持多种数据格式、数据容量较大等特点。

Firesheep反制工具BlackSheep

Firefox扩展程序Firesheep，能监听和劫持WiFi热点或无保护网络中的未加密社交网站账号。而现在Zscaler研究人员推出了反制工具BlackSheep。它也是Firefox上的扩展程序，能监视网络流量，如果发现网络中有Firesheep的活动痕迹，就会警告用户。

TaskAnt

专为个人/小团队协作、任务管理设计的计划工具，并为分配、跟踪、搜索任务进行了优化。通过TaskAnt可以把一个任务分配给团队成员，并像处理消息一样推进、评论任务，也可以通知发任务的人。目前，已支持最后期限设置，邮件提醒等功能。由国内团队开发。

愤怒的小鸟

Rovio Mobile在著名iPhone游戏“愤怒的小鸟”的流行之后，又推出了该游戏的周边产品，令人印象深刻的小鸟主角玩偶现在可以抱在怀中了！

GEEK产品

Sifteo Cubes

Sifteo两年前设计的概念游戏产品Sifteo Cubes将在今年推出。每一个小立方体重35克，尺寸4.3cm×4.3cm×1.9cm，完全充电可提供4个小时的运行时间。配备了32位处理器、128×128像素显示屏、3轴动作感应加速器、2.4GHz无线信号收发器、非接触式感应芯片，可通过无线连接电脑。每个立方体的屏幕可以根据不同的游戏来设计显示的内容。不同的立方体之间可以互动显示，可以感应碰撞，产生不同的游戏效果，理论上可以产生无穷多的组合。现在已经有了桌游、牌类、字母类、方块消除类、七巧板等游戏下载。有实力的话，甚至可以组成一套麻将。

C-Water

对于经常进行户外活动的朋友来说，饮水是户外非常重要且必不可少的。缺少水源时，如何取水将是很迫切的问题。这款设计充分利用了冷凝的原理，只需放置在潮湿的地面上，一段时间后，蒸发的水会在罩子顶部凝结然后流到旁边的容器中。

Quirky Perch

Quirky Perch是一个iPhone的扬声器底座，同时也可作为iPhone的充电插座。话筒通过蓝牙与iPhone连接。想法很简单，设计非常棒。

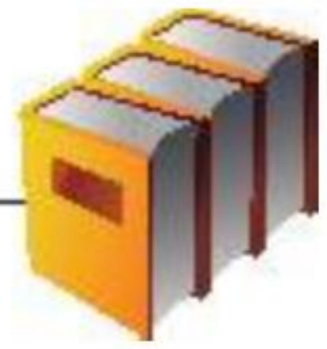
iTwin

不要被它的外形所欺骗，iTwin设计的这款产品虽然看起来像U盘，用起来也像U盘，但它其实不是U盘，而是一个用于远程文件共享和访问的USB设备，其容量大小取决于你的硬盘大小。每套产品有两个USB接口的设备，将这两个设备配对后，像U盘一样分别插到两台不同的电脑上、访问其中的文件。但是不同之处在于，这些改动会同时出现在另外一台插了这个设备的电脑中。这个小工具必须要联网才能工作，所有数据和通信都采用AES加密，相对于网络硬盘或云存储服务来说更加简单、安全。

Stealth超小型计算机

Stealth最近发布的超级迷你计算机，差不多仅有手掌那么大（102mm×155 mm×37mm），重0.54kg、CPU采用Intel双核1.9GHz或2.26GHz，最大支持8GB内存、500GB的2.5寸笔记本硬盘，支持SSD卡、Intel Mobile GM45Express芯片组，集成显卡千兆网卡，以及多种输入输出接口。支持Windows以及Linux系统。

新书上架



一步步写嵌入式操作系统——ARM编程的方法与实践

作者：李无言
出版社：电子工业出版社

Wintel体系正在瓦解，无论在平板还是手机领域，ARM无疑将越来越举足轻重。

本书作者就立足于嵌入式技术，以目前最流行的ARM体系结构为基础，为读者展示嵌入式环境下操作系统的基本原理和实现方法。

对于未曾奢想过能写操作系统的人来说，这本书将介绍怎样去实际编写一款嵌入式操作系统。

全书共分九章，从最基本的嵌入式编程方法开始，逐渐深入到中断管理、内存管理、设备管理、文件系统管理以及进程管理等操作系统核心部分，为读者系统地呈现了一个操作系统的全貌。

简约至上：交互式设计四策略

作者： Giles Colborne
译者：李松峰 秦绪文
出版社：人民邮电出版社

追求简单易用是人类的本性，无论是互联网产品，还是移动应用，抑或或其他交互式设计，简单易用始终都是赢得用户的关键。同时，简单易用的程度也与产品寿命的长短密切相关。可以举例的有，2007年便携式摄像机Flip在美国受到广泛追捧、2008年设计简约的日系汽车大行其道、2009年只容140字的Twitter风靡全球、2010年iPad甫一推出就引发抢购热潮，这些产品都以相当简约的设计风格引人注目。

在这本传授交互设计简约法则的小书（不足两百页，并且图文对页）中，作者依托20多年交互式设计的探索与实践，以警句式的写作风格提炼出四条交互设计中的金科玉律：合理删除、分层组织、适时隐藏和巧妙转移。

深入理解计算机系统（英文版·第2版）

作者：Randal E. Bryant、David R. O' Hallaron
出版社：机械工业出版社

本书从程序员的视角详细阐述计算机系统的本质概念，并展示这些概念如何实实在在地影响应用程序的正确性、性能和实用性。

作者写作的最大优点是为程序员描述计算机系统的实现细节，帮助读者在大脑中构造一个层次型的计算机系统。从最底层的数据在内存中的表示到流水线指令的构成，到虚拟存储器，到编译系统，到动态加载库，到最后的用户态应用。通过掌握程序是如何映射到系统上，以及程序是如何执行的，读者能够更好地理解程序的行为为什么是这样的，以及效率低下是如何造成的。

全书前部分的重点在于讲明硬件和软件如何有效/合理地相结合，后半部分关于系统，偏应用级的内容则择其大端，读者可以为后续学习打下坚实的基础。

设计原本

作者：Frederick P. Brooks Jr.
译者：InfoQ中文站 王海鹏 高博
出版社：机械工业出版社

无论是软件开发、工程还是建筑，有效的设计都是工作的核心。本书将对设计过程进行深入分析，揭示进行有效和优雅设计的方法。

本书包含了多个行业设计者的特别领悟。通过与几十位优秀设计者的对话，以及他自己在几个设计领域的经验，作者指出，大胆的设计决定会产生更好的结果。

作者追踪了设计过程的演进，探讨了协作和分布式设计，阐明了哪些条件造就了真正卓越的设计者。

同时作者探讨了设计过程的具体细节，包括多种预算约束条件、美学考虑、设计经验主义及工具。此外，他还将这些讨论与现实中的案例结合起来，这些案例从房屋建造到IBM的Operating System/360。总之，如何获得成功设计的关键因素贯穿全书，这是每一位设计者、设计项目经理和设计研究者都应该知道的。

MySQL技术内幕: InnoDB存储引擎

作者：姜承尧

出版社：机械工业出版社

作者从源代码的角度深度解析了InnoDB的体系结构、实现原理、工作机制，并给出了大量最佳实践，能帮助你系统而深入地掌握InnoDB，更重要的是，本书能为读者设计和管理高性能、高可用的数据库系统提供相当重要的指导。

全书首先介绍了MySQL独有的插件式存储引擎，分析了MySQL的各种存储引擎的优势和应用环境；接着逐一详细讲解了InnoDB存储引擎内部的各个功能模块，包括InnoDB存储引擎的体系结构、内存中的数据结构、基于InnoDB存储引擎的表和页的物理存储、索引与算法、文件、锁、事务、备份，以及InnoDB的性能调优等重要的知识；最后深入解析了InnoDB存储引擎的源代码结构，对大家阅读和理解InnoDB的源代码有重要的指导意义。本书适合构建和管理高性能、高可用性的MySQL数据库系统的开发者和DBA阅读。

HTML5高级程序设计

作者：Peter Lubbers 、Brian Albers 、Frank Salim、Ric Smith

译者：李杰 柳靖 刘淼

出版社：人民邮电出版社

推荐版HTML5规范预计将在2022年发布，这可不意味关注Web技术的人要等到那时才开始HTML5学习。事实上，HTML5一直不断进化、完善（HTML5的一些特性非常具有革命性）。可以说再不关注HTML5就Out了，而这本书将是一个很好的起点。

即使现在HTML5能做到的事情也比能想到的多。书中概括地介绍了HTML5规范的缘由、发展和现状，这对正面临技术选型选择的技术人员有相当引导作用。

作者们全面展示了如何使用WebSocket、Geolocation、Web Storage、Canvas及Audio/ Video等前所未有的新特性构建最炫的Web应用，并以大量的示例涵盖全部HTML5 API。此外，还介绍了当今浏览器对HTML5特性的支持情况。

名为“高级程序设计”，书中代码部分无疑面向需要进阶技术人员，但任何想一窥浏览器发展前景的读者都可以从中得到启发。

小小黑客之路: 黑客工具、攻防及防火墙编程入门

主编：葛垚

出版社：电子工业出版社

成为一名成功的黑客需要正确的指引，这本书正如一卷黑客攻防的探险地图，带领初窥信息安全门径的人们进入充满惊喜的黑客世界。全书由入门、进阶、高级和综合四个层次组成，以C/C++语言和Windows API为平台，配合各种案例编写而成。本书共分为四部分。入门篇首先简要地介绍了黑客、黑客工具及黑客编程中常用的辅助工具。进阶篇和高级篇，分别介绍了七种网络上热门的黑客工具的编写技巧。综合篇中实现了两个较为复杂的软件。附录A与附录B中介绍的网络协议和PE格式的基础知识是黑客编程的基础。

全球排行榜

	Amazon
1	Hackers
2	Computer Organization and Design, Fourth Edition: The Hardware/Software Interface
3	Head First HTML with CSS & XHTML
4	Introduction to Algorithms, Third Edition
5	Computer Networking: A Top-Down Approach (5th Edition)
6	Getting to Know ArcGIS Desktop
7	C Programming Language
8	MATLAB: An Introduction with Applications
9	Building Java Programs: A Back to Basics Approach (2nd Edition)
10	Operating System Concepts

	天珑书局（中国台湾）
1	Google Android 2.X应用程序开发实战
2	Google Android SDK开发范例大全 2
3	深入浅出Android系统原理及开发要点
4	iPhone创意程序设计家, 2/e
5	全球最强VMware vSphere 4企业环境建构
6	鸟哥的Linux私房菜——基础学习篇, 3/e
7	软件构筑美学：当项目团队遇上失控程序，最真实的解决方案
8	App程序设计入门——iPhone & iPad
9	详解Objective C——iPhone / iPad / Mac OS X标准程序设计与实作
10	Silverlight 4商业级应用程序开发

	第二书店
1	深入理解计算机系统（原书第2版）
2	ActionScript大型网页游戏开发
3	编程人生：15位软件先驱访谈录
4	MySQL技术内幕：InnoDB存储引擎
5	程序员的思维修炼：开发认知潜能的九堂课
6	Android应用开发揭秘
7	高效程序员的45个习惯：敏捷开发修炼之道
8	重构：改善既有代码的设计
9	程序员修炼之道——从小工到专家
10	HTML5揭秘

高嫄，安卓爱普公司创始人。原搜狐媒体产品中心经理。关注移动互联网、互联网产品管理。
微博：
httpt.sina.com.cn/inetpm

社会化电子商务：小蚁雄兵的新机会

家成立才两年的创业公司，居然拒绝了Google的60亿美元收购，太不可思议了！这个做优惠券生意的小公司凭啥值这么多钱？这是最新一期*Wired*杂志英国版2011年2月刊，主编在卷首语颇为“不平”写下的第一段话。这期*Wired*封面专题就是“你的社交网络如何影响了你的购物？”幸好没有按时下流行的套路，否则标题会是“电子商务已死，社会化电子商务永生”。

Groupon令人咂舌的60亿美元估值，并不是社会化电子商务即将爆发的唯一信号。Amazon对团购网站LivingSocial投资1.75亿美元，eBay以7500万美元收购本地购物搜索引擎Milo。这些电商大鳄们近期频频出手，正是因为意识到电子商务领域风雨欲来。Econsultancy的研究表明，人们90%以上的购买行为，都要受到社会化因素的影响。谁能够最好地利用社会化网络来影响、撮合、促成人们的交易行为，谁就可能成为未来赢家。

Yahoo!在2005年12月就提出了社会化电子商务，此概念并不新鲜，即使是当红炸子鸡Groupon模式的团购，也都不乏先例。在中国20世纪70年代，地方上的农村供销社就有人专门负责登记村民的购物需求，再到县里把商品运回来；20世纪80年代，温州桥头镇有好事者把家家户户的纽扣作坊联合起来，一起与原料供应商谈价。与做供应链系统的朋友聊，他第一反应脱口而出，“团购”不就是“集采”嘛。

为什么“社会化电子商务”能够一夜之间成为Buzzword？从Google Trend上看，社会化电子商务在2010年7～8月间达到关注度的高峰，这正是Facebook历史性地达到5亿用户的时段。以Facebook和Twitter为代表的社会化媒体的成熟，使得社会化电子商务在规模、深度上都拥有过去无法想象的可能性空间，也涌现了团购、限时抢购、社会化购物分享网站、购买记录分享网站、整合LBS的社会化购物、个人购物助理等多样化的探索方向。

社会化电子商务，对成百上千万小蚁雄兵式的中小商家可能意义更大，是他们得以借力的新草根式营销和商务平台。新浪广告刊例每季度都在涨，百度的竞价排名每天都在涨，就连成长迅速的360网址站每月也都要涨一次价。甚至依靠海量C2C卖家起家的淘宝，也通过调整商品搜索规则、推出淘宝商城等一系列措施，游戏规则越来越向大卖家、商城卖家一侧倾斜，逐渐冷落和抛弃了广大的中小卖家。现在的社会化电子商务，正处于蛮荒开拓阶段，一如早期做百度SEO、2007年之前做淘宝卖家，是一个难得的机会空窗期。

当年，搜索引擎的出现彻底颠覆了传统广告业的大腕俱乐部模式，搜索引擎的竞价广告以其廉价、精准、大众化吸引了被传统广告业所排斥的、本来做不起广告的中小商家甚至个人。现在，社会化电子商务则更进一步，正如*Fast Company*杂志所说：在未来，营销就和性一样，只有没本事的才需要花钱去获得。最新的一份报告显示，全球垃圾邮件的数量第一次出现了比较显著的下降，推测可能是转向了社会化媒体的营销方式。垃圾邮件发送者虽然可恶，但他们嗅觉灵敏、逐水草而居，这可以作为一个风向标，供机动敏捷、灵活适应的中小商家们思索参考。

社会化电子商务，营销不需要花钱去获得，但需要用心去获得。中小商家在利用社会化电子商务时，更能发挥自己与大商家、品牌商家的区别优势，凸显生意的人性一面，保持与顾客的私人化、人情化关系。如可避免“安利效应”（当你向朋友推销，你也就失去了朋友）；可借鉴“沉没成本”心理（当购买并拥有一个产品之后，顾客心态就由怀疑转变为支持）。人们通过购物和消费来进行“自我价值表达”，如何强化和帮助人们希望向自己的社交网络传达的自我形象，如何通过关系和沟通来快速有效地传播和放大，从而引起类似品位和心理的人们的效仿和追随行为，这可不仅仅是在宝贝标题中加“全民疯抢”那么简单。

以创新方式更好地服务活力蓬勃、野蛮生长的中小商家，Google是这样成功的，淘宝曾经是这样成功的。而社会化电子商务正孕育着足以挑战Google们的伟大企业！

王煜全，Frost & Sullivan中国区总裁。

由Android吸费手机想到……

2010年底，大家欢欣鼓舞回顾一年的成绩时，一个坏消息也在传播：基于开放操作系统Android平台的手机应用出现了吸费现象，而且似乎比例很高。一些应用软件在用户不知情的情况下，通过短信等功能，将用户的费用悄悄吸走。业内惊呼，当年的SP又开始祸害新的移动互联网产业了。

为什么会出现吸费情况

在Android平台上出现这种情况，主要是两个条件造成的。一是开放的操作系统，Android系统可对终端很多功能提供开放式调用，其最初目的是让开发者充分利用手机能力。二是在开放的环境下，软件应用如何获得收益。说到开放，大家就会想到免费，那么开放的Android系统上的应用，很多用户当然也希望是免费的（实际很多正在免费）。这种情况下，开发者的利益只能够从广告模式上考虑，但广告的收益浮动很大，并且软件应用中的广告仅是一种可选形式，不能够满足应用提供者的收益要求。

于是，一些短视、激进的应用提供商开始想其他“办法”。目前的吸费方式基本是通过调用用户的短信权力，在其不知情的情况下，以用户的名义发送一些付费的定制信息，实质和以前SP在WAP网页上设置的点击收费方式如出一辙。

如何消除吸费现象

技术手段上阻止

出现了吸费现象，首先要想办法阻止这种损害用户利益的情况继续蔓延。Android手机吸费事件曝光相对较早，在用户中还没形成原来梦网SP恶意扣费那么严重的影响，因此，快速制定措施来堵住源头是当务之急。

近日，中国移动宣布推出“业务扣费主动提醒”和“增值业务‘0000’统一查询退订”两大免费服务，就是针对这些现象的应对措施。这也是运营商首次推出扣费主动提醒业务。只要在用户费用支出前通知到用户，让用户来做主，就杜绝了用户不知情的恶意扣费。

同时，系统也应该对短信等相关能力的调用进行规范，不容许随意调用可能产生扣费途径的手机功能。

产业环境上合作

在整个产业链上，应该为产业的协调发展创立良好的环境，杜绝个别非法行为对整个产业的影响，这就需要我们建立一套有效的信用体系共享机制。

比如在开放企业功能和终端能力方面，我们相信开放调用一定会有违反规则的，比如利用新浪微博的开放，做违法信息传播，不但新浪微博受害，整个产业也会受害。开放企业进行合作，应当提供应用开发者信用共享机制，如果利用新浪微博的开放能力违规，就把违规者记录下来，并且通知所有其他提供开放API的企业，大家对其形成统一的封杀。这样，Android平台上，通过应用软件恶意吸费的应用得到及时通报，对违规企业形成及时隔离，增加它的违规成本。从而形成一个信用共享的信用体系，良性发展，应用就会繁荣。

业务应用上创新

无线互联的发展需要创新，而且需要突破性创新、颠覆性创新。目前我们看到的创新基本属于面向终端、面向眼前的应用创新，这些创新只是满足了目前部分用户在一定时间内的部分需求，其实充满了很大的不确定性。例如，面向手机的浏览器、通讯簿、桌面管理工具等。这些应用的特点是：面向眼前甚至是过去的产品开发；大公司很容易进行模仿，只要它愿意；开发者众多，竞争激烈。这些面向现在的应用使得开发者没有很好的收费点。假如我们能够在业务上形成突破性的创新，设计出用户必需的应用，设计出面向未来的应用，那么就会形成良好的收入机制，实现颠覆传统的创新。

Robbin:
Unix程序员,AppleFans。Geek,急需女朋友。命令行狂热者。

Hank:
Windows程序员,.NET高手,已婚,保守派。

Ada:
架构师。第三性别，喜欢挖苦男人。热爱无所不能的emacs。

作者介绍：西乔
项目经理，06年起携创NI团队从事Web技术外包开发及产品咨询顾问。

如果你有什么好玩的关于程序员的故事、对话、代码，愿意通过漫画的形式分享，请给西乔发邮件：arthur369@gmail.com。

幽　默

1. **产品经理**：我周一就在Bugzilla里面提交Bug给你，这都周五了，怎么还没改好？
   开发工程师：别急，让Bug飞一会儿。

2. 据说在每一个互联网公司里，都有一个扫地的老太太。很偶然地，当她经过一个程序员的身边，扫一眼屏幕上的代码，会低声提醒对方说："小心，栈溢出了。"

3. 楚国人坐船渡河时，剑不慎掉入江中，他在舟上刻下记号说："这是剑掉下的地方。"当舟靠岸时，他跳入河中把剑捞了上来，旁边的人诧异非常。楚国人淡定地说："云标记……"

4. 在某网站上看到一软件介绍。
   软件名称：瑞星小狮子中文版。
   软件语言：中文简体。
   软件类型：国产软件。
   运行环境：XP/2003/Vista。
   软件大小：5MB。
   软件简介：去掉了杀毒和防火墙功能，只保留小狮子。

5. 将来最终华为的竞争对手会变成一个公司，它的名字就是思科诺基亚西门子摩托罗拉阿尔卡特朗讯爱立信北电。

6. 学计算机的过人之处：
   一般人十个手指只能数到10，学计算机的能数到1024！

7. 程序员去一家咖啡店吃午饭，柜台里边的美眉问吃哪种面包。程序员不假思索地说："默认的。"

8. 一老外过马路的时候遇上红灯了，中国朋友意欲前行，老外叫住他："灯，等灯等灯！"中国朋友回过头来说："就你有英特尔啊！"

9. 史上三大吐血界面：
   - SAP GUI：完全不知道能做什么；
   - Photoshop：知道能做啥，就是不会做；
   - MS Office：知道怎么做，就是找不到对应的功能藏在哪儿。

10. **记者**：你选择拜访中国的动机和原因是什么？
    Facebook CEO扎克伯格：据我所知中国最大的几个网络社区，如天涯和豆瓣等，每天都有大量的用户在跟帖中只会写上两个字，"马克"，我是感受到召唤而去的。